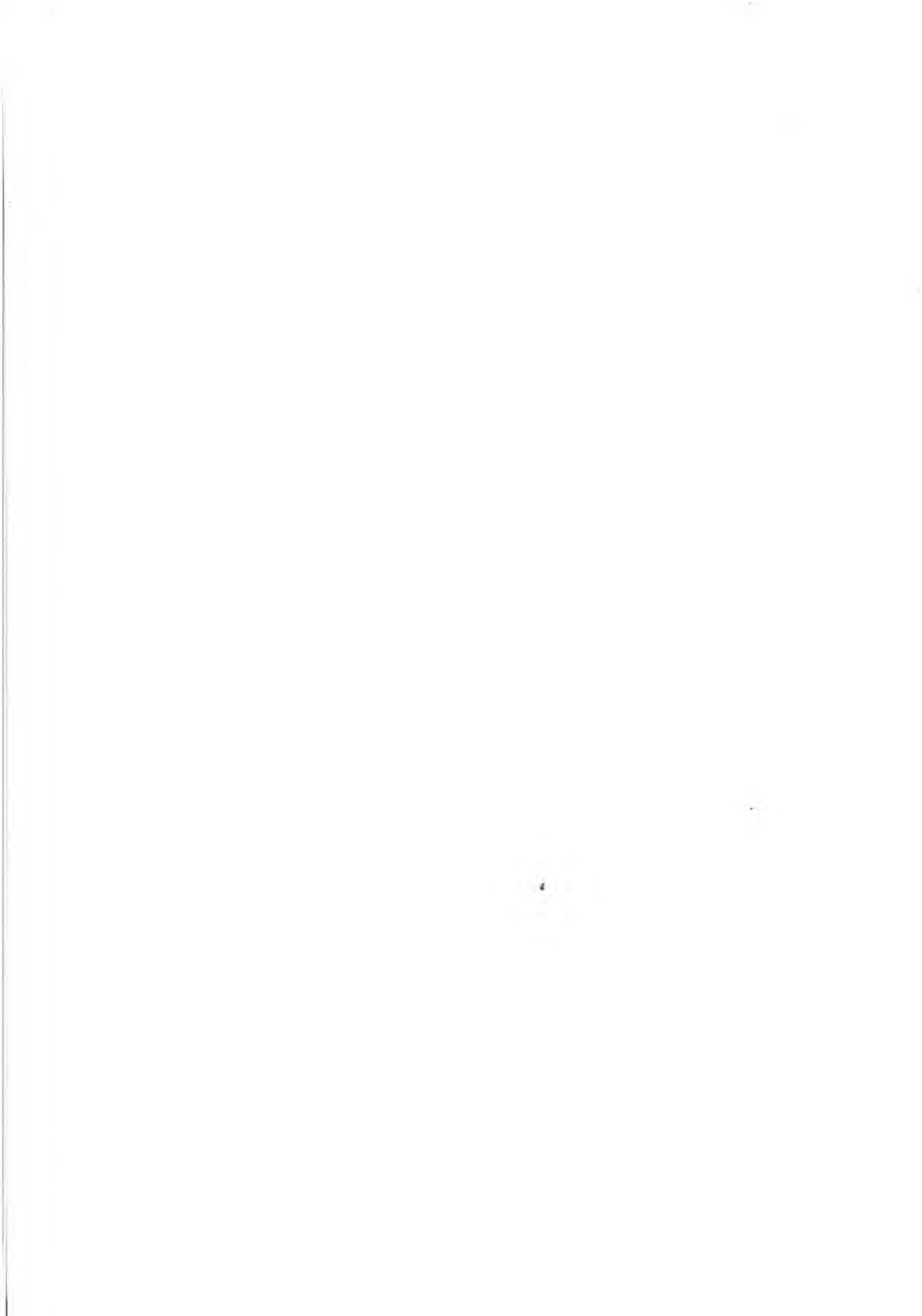
USER'S MANUAL

パソコンテレビ X1 turbo
パーソナルコンピュータ
形名
CZ-856C

上手に使って上手に節電

製造番号は、品質管理上重要なものですから商品本体に表示されている製造番号と、保証書に記載されている製造番号とが一致しているか、お確かめください。

■USER'S MANUAL(ユーザーズマニュアル)

はじめに

このマニュアルはＢＡＳＩＣの初歩から、本機の特長であるグラフィック機能、日本語処理機能、デジタルテロッパー機能などを使いこなしていただくため項目別に説明しています。

目的に応じて各章をお読みください。

入門編…パソコンは初めてという方のため、簡単なプログラムを中心に説明しています。

応用編…入門編が理解でき、さらに使いこなしていただくために説明しています。

なお、本機を正しくご使用いただくため、まず取扱説明書からお読みください。

◎ディスクＢＡＳＩＣ(ＣＺ-８ＦＢ０２)を起動させた場合

ディスクＢＡＳＩＣが起動されると自動的にＳｔａｒｔ　ｕｐというプログラムが実行され、各種の初期設定が行なわれます。

初期設定には、ディスクＢＡＳＩＣが起動されるときにすでに行なわれているものと、Ｓｔａｒｔ　ｕｐプログラム中で行なわれているものがあります。

ディスクＢＡＳＩＣ起動時の初期設定は変更できませんが、Ｓｔａｒｔ　ｕｐプログラム中で行なわれるものは、このプログラムを自分の希望するように書き換えておけば、初期状態を自由に決めることができます。

(ｉ) ディスクＢＡＳＩＣ起動時の初期状態

ＩＮＩＴ

ＷＩＤＴＨ　８０，２５，　ＣＯＮＳＯＬＥ　０，２４，０，８０

(高解像度ディスプレイモードのとき)

ＷＩＤＴＨ　８０，１２，　ＣＯＮＳＯＬＥ　０，１１，０，８０

(標準ディスプレイモードのとき)

ＫＭＯＤＥ１

ＯＰＴＩＯＮ　ＳＣＲＥＥＮ１

ＣＬＥＡＲ　＆ＨＦ４００

ＣＬＩＣＫ　ＯＮ

ＲＥＰＥＡＴ　ＯＮ

(ii) Ｓｔａｒｔ　ｕｐで設定するもの

ＣＬＥＡＲ　＆ＨＦ０００ (音訓辞書使用時)

ＫＬＩＳＴ　１

ＦＵＮＣＴＩＯＮ　ＫＥＹの設定

内蔵時計の「年」の項の設定

(iii) ＮＥＷＯＮの設定

ＮＥＷＯＮ命令はＢＡＳＩＣのコマンドやステートメントなどを削り、フリーエリアを増やします。

ディスクＢＡＳＩＣを起動した直後、

```
ＮＥＷＯＮ■
        ↑ カーソル
```

と画面に表示されます。
0～9までの数値を入力するか、または数値を省略してリターンキー（↵）を押してください。希望のレベルのＮＥＷＯＮが設定できます。

```
NEWON ↵
```

とキー入力すれば、すべてのコマンド、ステートメントが使用できます。
ＮＥＷＯＮのレベル（0～9）については、応用編「6章 5 ＮＥＷＯＮとフリーエリア」をお読みください。

◎従来のＸ１用ＢＡＳＩＣ（ＣＺ-８ＣＢ０１、ＣＺ-８ＦＢ０１）とＸ１turbo用ＢＡＳＩＣ（ＣＺ-８ＦＢ０２）との初期設定の相違点

ＣＺ-８ＣＢ０１、ＣＺ-８ＦＢ０１	ＣＺ-８ＦＢ０２
WIDTH 40,25	WIDTH 80,25（高解像度ディスプレイモードの時）
WIDTH 40,25	WIDTH 80,12（標準ディスプレイモードの時）
CONSOLE 0,25,0,40	CONSOLE 0,24,0,80（高解像度ディスプレイモードの時）
CONSOLE 0,25,0,40	CONSOLE 0,11,0,80（標準ディスプレイモードの時）
KLIST 0の状態	KLIST 1
KMODE 0の状態	KMODE 1
CLEAR &HFF00	CLEAR &HF400

Ｘ１turbo用ＢＡＳＩＣは、以上の点で、従来のＸ１用ＢＡＳＩＣと異なっていますので、Ｘ１turbo用ＢＡＳＩＣ上で従来のＸ１用のソフトウェアを実行する際には、次の操作を行なってください。

```
CONSOLE 0,25:KLIST0:KMODE0:CLEAR &HF400 ↵
```

ただし、Ｘ１用のソフトで＆ＨＦ４００以降を使用しているソフトは利用できません。
なお、本機にはＸ１用ＢＡＳＩＣ（ＣＺ-８ＦＢ０１ Ｖ１.０、ＣＺ-８ＣＢ０１ Ｖ１.０）が搭載されています。これの起動法については『アプリケーションソフトの説明書』の「はじめに」の項をお読みください。

ユーザーズマニュアル

14章　マウスインターフェイスの使い方

15章　デジタルテロッパーの使い方

付録

第１部

入門編

1章
プログラミングを始めよう

1 コンピュータへのメッセージ

本機に電源を入れＢＡＳＩＣをロードします。
（取扱説明書第２章の「動かしてみる」を参照）
ディスクＢＡＳＩＣを起動すると、ＮＥＷＯＮ■と表示します。（図１）
ＮＥＷＯＮ■　と表示したら⏎（キャリッジリターンキー）を押してください。
すると、図２のようになります。
画面に四角くて明滅するものがあります。
このチカチカするものを**カーソル**といいます。カーソルは表示位置を示し、コンピュータがあなたからのメッセージを待っています。

```
SHARP HuBASIC  CZ-8FB02 Version1.0
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P


NEWON■
```

図１

```
SHARP HuBASIC  CZ-8FB02 Version1.0
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
 74493 Bytes free

NEWON
Ok
■
```

図２

では、手はじめに、キーを
Ｃ　Ｌ　Ｓ
の順に押してください。画面は図３のようになります。

```
Ok
CLS■
```

図３

画面には大文字で表示されていますが、小文字を使ってもかまいません。
（大文字を使うときは CAPS LOCK キーを押してロック状態にします。）

次に、⏎（キャリッジリターンキー）キーを押してみてください。画面に書かれていた文字がすべて消え、図４のように表示されましたね。

```
Ok
■
```

図４

"**ＣＬＳ**"には「画面をきれいにしてください」という意味があります。ＣＬＳのようにコンピュータが理解する言葉を**キーワード**といいます。キーワードは全部で２００以上あります。いっぺんには覚えられませんから、少しずつ覚えるようにしましょう。

また、キーワードの最後に⏎キーを押してください。これをしないとコンピュータは何もしてくれませんので忘れずに行ってください。

それでは次に、**PRINT**というキーワードについてみてみましょう。

PRINT"X1"

と打って⏎キーを押して画面をみると図5のようになっていることがわかりますね。

PRINT"23+44"

と打って⏎キーを押すと、図6のように表示されます。

これは、PRINTが「引用符（"）で囲まれた文字を画面に表示しなさい」という意味をもったメッセージだからです。

図5

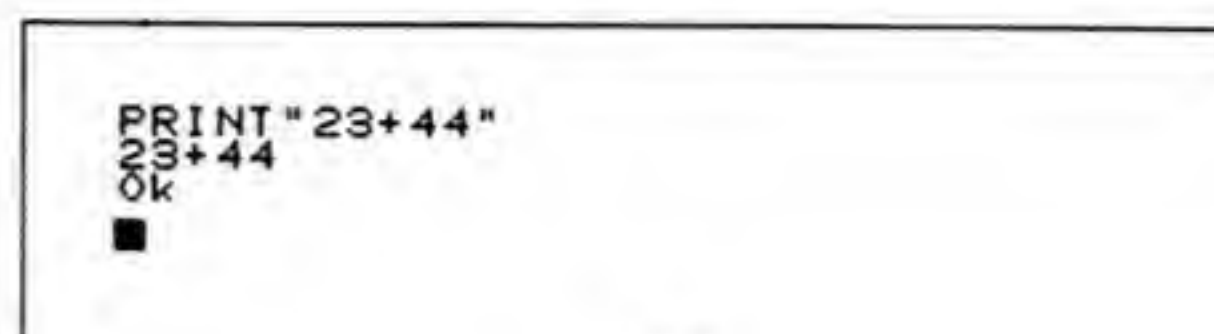



図6

それでは、次に示す例を順に試してください。⏎の記号は「その位置で⏎の記号を打ってください」という意味です。

（例1）PRINT　23+44⏎
　　␣67
　　Ok
　　■

（例2）PRINT（23+44）*2／5⏎
　　␣26．8
　　Ok
　　■

（例3）？23+44⏎
　　␣67
　　Ok
　　■

上の例は（例1）が23＋44のたし算を、（例2）が（23＋44）×2÷5の計算をそれぞれコンピュータにさせ、その結果を表示させたものです。

これは、PRINTに「引用符（"）で囲まれた文字を表示しなさい」という意味以外に「計算した結果を表示しなさい」という意味もあるので、それを利用したものです。

なお、（例3）だけはちょっと別のキーワードの例のようですが、実はPRINTは？（疑問符）で置き換えて使うことができます。キーボードからP、R、I、N、Tと5文字入れるところが？の1文字で済むのでこれはたいへん便利ですね。

これまでみてきたように、キーワードは、その名の通り、あなたがコンピュータにメッセージを伝えるための鍵となる単語で、キーボードから正しく打ち込んで⏎キーを押すと、コンピュータはそのメッセージに応じた仕事をしてくれます。

2　まずは簡単なプログラム作りから

それでは、

PLAY⏎

と打ち込んでください。

Missing operand

```
Ok
■
```

というメッセージが返って来ました。

これは、コンピュータが「ＰＬＡＹというキーワードの意味はわかるけれども、ＰＬＡＹの後ろに必要なデータが書かれていないので何もできません」と途方に暮れているのです。その必要なデータをいっしょに打ち込んでみましょう。

ＰＬＡＹ"ＢＧ"↵

さあ、どうなりましたか？何にも起こらないという人は本機の音量調整を上げてもう一度打ち込んでみてください。「シー・ソー」という電子音が出るはずです。

ＰＬＡＹは音を出すためのキーワードで、音を表わす文字を引用符（"）で囲んで指定すると、音を出すことができます。ここでは、Ｂ（シの音）とＧ（ソの音）を指定したので「シー・ソー」という音が出たのです。これは、さきほどのＰＲＩＮＴと使い方が似ています。

ＰＲＩＮＴ"ＣＨＩＭＥ"↵

打ち込むと、引用符で囲んだ文字ＣＨＩＭＥが表示されますね。

では、次のように打ち込むとどうでしょう？

10 PRINT"CHIME" ↵

何ごとも起こりませんね。「ＣＨＩＭＥ」と表示しないし、「Ｓｙｎｔａｘ　ｅｒｒｏｒ」などのメッセージが返って来るわけでもありません。

今度は、

20 PLAY"BG" ↵

と打ち込んでください。やはり何ごとも起きません。でも、次のように打ち込むとどうなりますか？

ＲＵＮ↵

画面に「ＣＨＩＭＥ」と表示されて、「シー・ソー」という音が出たでしょう。

ＲＵＮと打ち込んで、↵キーを押してやれば、何度でも同じことを繰り返すことができます。いったいどういうわけでしょう。

ＬＩＳＴ↵

と打ち込んでみてください。

```
10 PRINT"CHIME"
20 PLAY"BG"
Ok
■
```

と画面に表示されるでしょう。

ＰＲＩＮＴやＰＬＡＹの前に１０とか２０の番号をつけるかつけないかでコンピュータの反応が違っています。

番号をつけないで、キーワードを打ち込んで↵キーを押すと、コンピュータはすぐその場で実行して、画面に文字を表示したり、音を出したりします。

番号をつけると、コンピュータはすぐに実行しませんが、打ち込まれたメッセージをきちんと覚えているので、あとからＲＵＮ↵と打ち込めば何度でも実行することができます。ここで、

```
10 PRINT"CHIME"
20 PLAY"BG"
```

の２行のことを**プログラム**といい、１０や２０の番号のことを**行番号**、行番号の後ろのＰＲＩＮ

T"CHIME"やPLAY"BG"のことを**ステートメント**といいます。

また、プログラムを実行するために打ち込んだRUNというキーワードのことを**コマンド**といいます。コマンドは、プログラムをどうするかコンピュータに指示するときに使います。アンダーラインのついた言葉は大切ですのでよく覚えるようにしてください。

RUNは「プログラムを実行しなさい」

LISTは「プログラムを画面に表示しなさい」

CLSは、「画面に書かれた内容を消しなさい」

という意味のコマンドです。

LIST⏎

と入れると、画面にプログラムが表示されます。

RUN⏎

と入れると、プログラムを実行します。

CLS⏎

と入れると、画面がきれいになりますが、プログラムもいっしょにきれいに消えてしまったかというと決してそうではありません。その証拠に、

LIST⏎

と入れると、プログラムが表示されます。

3 プログラムの仕組み

コンピュータには、命令を入力すると、電卓のようにすぐ答えが表示される**直接実行**と、命令を入力してもRUN⏎を押さないかぎり答えを表示しない**間接実行**があります。一般に、間接実行のことを"プログラム"と呼んでいます。

3.1 直接実行と間接実行のちがい

間接実行させるには行番号を各行のあたまに付けてコンピュータに順序よく仕事をさせるようにしますが、行番号を付けないで命令を入力するとすぐに実行され結果が表示されます。これが直接実行です。

たとえば4＋5の計算をさせたいとき、右の上の行のようにするとすぐ答の9が下側に表示されますね。これが直接実行です。

```
PRINT 4+5
 9
Ok
■
```

こんどは行番号を付けて右のように入力します。このときは画面に結果があらわれません。

```
10 PRINT 4+5
└─(これを行番号と呼びます。)
RUN
 9
Ok
■
```

ここでRUN⏎と入力するとはじめて答の9が表示されます。

これが間接実行と呼ばれるもので行番号を付けることによってその内容は「プログラム」として扱われすぐに実行されませんが、何度も繰り返し実行させたり、順序よく実行させることができます。

3.2 プログラムって何だろう

間接実行によってコンピュータに仕事をさせようとする場合、「プログラム」が必要になります。

プログラムというのは、コンピュータにどういう順序で仕事をさせるかを書いた手引書のようなもので、コンピュータはプログラムに書かれている行番号の小さい順に仕事をこなして（実行して）いきます。

たとえばコンピュータに1から15までの数字を画面に表示させるプログラムを書いてみると次のようになります。（ここから⏎キーの表示は記していません。）

```
10 FOR I=1 TO 15
20 PRINT I
30 NEXT I
40 END
```

これがプログラムです。

プログラムを実行させた画面

ここで、RUN（プログラムを実行せよ）と入力すると行番号10、20…の順にコンピュータは実行していきます。40番を実行してプログラムの実行が終了します。このようにプログラムは基本的には若い行番号から順に実行されます。

3.3　変数を使う

プログラムの中では、いろいろな数値を扱いますが、値の決まっていないものやいろいろと値が変わるものは**変数**という形で扱うと便利です。
たとえば次のプログラムは、たし算をさせようとするプログラムです。この中のAやB、Cが変数で、アルファベットまたはアルファベットと数値の組合せで名前を付けます。ただし変数の名前は240字以下で最初の1文字はアルファベットでなければなりません。

```
10 INPUT A,B
20 C=A+B
30 PRINT C
40 END
```

このプログラム中のAやB、Cが変数です。AやBはいろいろ値を変えることができます。CはA＋Bの計算結果で値が変わりますのでこれも変数です。

一般に使用している数値にも整数や小数、分数などがあるようにコンピュータで使うことのできる変数にも、整数型、単精度実数型、倍精度実数型、文字型などの区別があり用途に応じて使い分けます。

3.4　困ったとき助けてくれる CTRL ＋ D キー

本機を使用されて行く途中、次表のようなことで困ったことが生じるかも知れません。これらは無意識のうちにコンピュータをそのような状態にさせた結果ですから、次のような命令で正常にもどしてください。

CTRL キーを押しながら D キーを押して、
そのあと
SHIFT キーを押しながら CLR HOME キーを押します。

1	キー入力された文字がすべて点滅文字になってもとにもどらないとき。
2	キー入力された文字がすべて反転文字になってもとにもどらないとき。
3	キーから入力された文字の内容が、キーの表示内容と合わなかったりわけのわからないパターンが出てきたとき。
4	キーから入力した文字が標準の大きさでなく、倍文字となるとき。
5	キーから入力した文字の色を白色にもどしたいとき。
6	SHIFT ＋ CLR HOME キーを押しても画面の文字が消えずに残っているとき。
7	４０文字モードで、キー入力しても画面に何もあらわれないとき。
8	コンピュータの音が鳴りっぱなしで止まらないとき。
9	画面のグラフィックがＣＬＳ０の命令でも消えないとき。
１０	キー入力した文字がグラフィックのうしろにかくれてしまうとき。
１１	グラフィックで描いた図形が指定した色と合わないとき。
１２	ＬＩＳＴ，ＦＩＬＥＳコマンドなどによって、グラフィック画面が表示されなくなったとき。

4　プログラムの作成(その１)

4.1　プログラムの入力

プログラムをキーボードから打ち込んでコンピュータに覚え込ませることを、プログラムを**入力する**といいます。

さきほど入力したプログラムはたった２行でしたが、何十行、何百行となるのが普通です。そういった大きくて複雑なプログラムを入力する際には必ず途中で追加、変更、削除などの編集を行わねばなりません。

ここでは、小さなプログラムを例にとり、プログラムの編集方法を具体的に説明しましょう。

まず、ＮＥＷというコマンドを使って、コンピュータ内に残っているプログラムをきれいに消しましょう。

　　ＮＥＷ↵

試しに、

　　ＬＩＳＴ↵

と打ち込んでみてください。

　　Ｏｋ

　　■

と表示されて、プログラムがきれいに消えていることがわかります。

次に画面をきれいにしましょう。

　　ＣＬＳ↵

さあ、プログラムを入力する準備は整いました。スペルを間違えないように、プログラムを入力して行きましょう。

```
10 PRINT"X1" ↵
```

プログラムを１行入力したら、まず正しく入力されたかどうか見なおします。間違いが見つかったら、面倒でももう一度入力しなおしてください。何も全部入力しなおさなくても、もっとよい方

法があるのですが、それについてはあとで説明します。

正しく入力されたことを確かめて、

```
RUN⏎
X1
Ok
■
```

と表示されたらオーケイ。もし、

```
Syntax error in 10
Ok
■
```

とメッセージが出たら、

```
10 PRINT"X1" ⏎
```

と再び入力しなおしましょう。

20 PLAY"BGADDR" ⏎と入力し、さらに

LIST⏎と入力すれば

```
10 PRINT"X1"
20 PLAY"BGADDR"
Ok
■
```

と表示されます。

これでプログラムが2行になりました。実行してみましょう。

```
RUN⏎
X1
```

と表示されて音が出るでしょう。

```
30 PLAY"DE#FGGR" ⏎
LIST⏎

10 PRINT"X1"
20 PLAY"BGADDR"
30 PLAY"DE#FGGR"
Ok
■
```

これでプログラムが3行に増えました。RUN⏎と入力して、このプログラムを走らせる（RUNの本来の意味は「走る」です）と、「X1」と表示して、さきほどより長いメロディーが流れます。

4.2　1行の挿入

```
15 PRINT"CHIME" ⏎
LIST⏎

10 PRINT"X1"
15 PRINT"CHIME"
20 PLAY"BGADDR"
30 PLAY"DE#FGGR"
Ok
■
```

１０と２０の間に新しい１５の行が**挿入**されて、プログラムが4行に増えました。

RUN⏎

X１

CHIME

上のように表示されて、音が流れましたか？

プログラムは基本的には若い行番号から順に実行されます。ここでは、まず行番号１０のPRINT、次いで１５のPRINT、そして２０と３０のPLAYの順に実行されます。

4.3　1行の削除

それでは、X１と表示しないようにするにはどうすればよいのでしょうか。最も簡単な方法はプログラムから１０番の行を取ってしまえばよいのです。そのためには、行番号のうしろに何も入力しないでリターンキーを押せばよいので、

10 ⏎

と入力します。

LIST⏎

```
15 PRINT"CHIME"
20 PLAY"BGADDR"
30 PLAY"DE#FGGR"
Ok
■
```

１０番が削除されて3行のプログラムになっていることがわかります。走らせると、

RUN⏎

CHIME

と、「X１」を表示しないでメロディーが流れます。

このように、プログラムから１行削除するには、その行番号だけを入力して⏎キーを押せば済みます。

4.4　1行の変更

３０のPLAYの音を変えてみましょう。３０のステートメントを新たに入力すればよいのです。

30 PLAY"DABGGR" ⏎

LIST⏎

```
15 PRINT"CHIME"
20 PLAY"BGADDR"
30 PLAY"DABGGR"
Ok
■
```

前の３０番の行が、いま入力した内容に**変更**されました。さっそくプログラムを走らせてみましょう。どんな音が出るのかな？

RUN⏎

もし「Syntax error」が出たら、もう１度入力しなおすことを忘れずに。

5 CLS ⏎

LIST⏎

```
5 CLS
15 PRINT"CHIME"
20 PLAY"BGADDR"
30 PLAY"DABGGR"
Ok
■
```

CLSは、「画面をきれいにする」という意味のコマンドですが、プログラムのステートメントとしても使うことができます。

RUN⏎

画面をきれいに消してから「CHIME」と表示して音がでるようになりました。

4.5 行番号のつけなおし

RENUM⏎

Ok

■

あれっ、何をしたのかな？

LIST⏎

```
10 CLS
20 PRINT"CHIME"
30 PLAY"BGADDR"
40 PLAY"DABGGR"
Ok
■
```

よく見るとプログラムの行番号が変わったのがわかります。でも、プログラムの内容まで変わったわけではないので安心してください。

RUN⏎

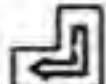
と入力して走らせると、コンピュータは、前と同じことをやってくれるでしょう。

RENUMは、「プログラムの行番号をつけなおしなさい」という意味をもつコマンドです。このコマンドは、プログラムに新しい行を挿入するために行間のスペースがなくなったときに使うと便利です。

さて、プログラムを増やしましょう。

```
50 PLAY"GBADDR" ⏎
60 PLAY"DABGGR" ⏎
70 PRINT"END" ⏎
```

プログラムが正しく入力されたかどうかを確かめてから、走らせてみましょう。

LIST⏎

```
10 CLS
20 PRINT"CHIME"
30 PLAY"BGADDR"
40 PLAY"DABGGR"
50 PLAY"GBADDR"
60 PLAY"DABGGR"
70 PRINT"END"
```

Ｏｋ

ＲＵＮ⏎

どうですか。チャイムらしい音が出ましたか。

このプログラムはいま先頭の番号が１０になっていますが、ＲＥＮＵＭを使ってプログラムの先頭の行番号を変えることができます。たとえば、先頭を１００にしたいときは

ＲＥＮＵＭ１００⏎

Ｏｋ

■

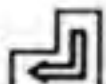
と入力します。

ＬＩＳＴ⏎

```
100 CLS
110 PRINT"CHIME"
120 PLAY"BGADDR"
130 PLAY"DABGGR"
140 PLAY"GBADDR"
150 PLAY"DABGGR"
160 PRINT"END"
Ok
■
```

試しに走らせてみると、さきほどと同じプログラムであることがわかるでしょう。

ＲＵＮ⏎

４．６　複数行の削除

さて、今度はこのプログラムを消すことを考えてみましょう。残らず消してしまうにはＮＥＷというコマンドを使えばよいのですが、もし１００～１６０まであるプログラムのうち１１０～１５０の行を消したいときにはどうすればよいのでしょうか。

消したい行の行番号を１つ１つ入力して行くのもよいでしょうが、面倒だという人のためにもっとよい方法があります。それは、ＤＥＬＥＴＥというコマンドを使う方法です。このコマンドを使えば、プログラムのいらなくなった行をまとめて消してしまうことができます。

１１０～１５０の行を消したいときは次のように入力します。

ＤＥＬＥＴＥ１１０－１５０⏎

これで、コンピュータは１１０～１５０のプログラムをすべて消してしまいました。確かめるため、プログラムを見てみましょう。

ＬＩＳＴ⏎

```
100 CLS
160 PRINT"END"
Ok
■
```

このプログラムを走らせると、画面が消えて「ＥＮＤ」と表示されるはずです。

ＲＵＮ⏎

ＥＮＤ

Ｏｋ

■

ここで、これまで説明したプログラムの編集方法について1度まとめておきましょう。

❖ プログラムの編集方法（その1） ❖

1. 1行挿入するには

⇨挿入したい行の上と下の行の行番号の間の番号を行番号に選ぶ。

たとえば、行番号２０と３０の間に挿入したいときは２１～２９の行番号で始まる1行を入力する。

2. 1行削除するには

⇨削除したい行の行番号のみを入力する。

3. 何行かまとめて削除するには

⇨ＤＥＬＥＴＥコマンドを使う。

たとえば、行番号１１０～１８０を削除したいときは、

ＤＥＬＥＴＥ１１０－１８０⏎

と入力する。

4. 1行変更するには

⇨変更したい行の行番号で始まる1行を改めて入力する。

---

5 プログラムの作成(その2)

次に進む前に、プログラムと画面をきれいに消しておきましょう。

```
NEW⏎
Ok
■
CLS⏎
Ok
■
```

5.1 カーソルの移動と文字の修正

ここでは、新たに簡単な曲を演奏するプログラムを入力してみましょう。

```
10 PLEY"CRCRGRGRARARGGRR" ⏎
```

ここでＰＬＡＹと入力するところを誤ってＰＬＥＹと入力したことに気づいたとします。

こういう場合、どうやって誤りを訂正したらよいでしょうか。もう1度「１０…」と行番号１０から入力しなおすのも一つの手ですが、もっとよい方法があります。

それは、キーボードのテンキー部分の下方にある**カーソルコントロールキー**を使う方法です。

カーソルコントロールキーは矢印のついたキーで、⇧⇩⇦⇨の4つあり、画面上のカーソルを上下左右好きな方向に動かすことができます。

ＰＬＥＹの「Ｅ」を「Ａ」になおすためには、まずカーソルコントロールキーを使ってカーソルをＰＬＥＹの「Ｅ」のところまでもって行きます。

```
10 PLEY"CRCRGRGRARARGGRR"
■
↑
カーソル
```

の状態で、⇧を1回、⇨を5回打ってみてください。カーソルが、ちょうど「Ｅ」のところに来たらオーケイです。もし、「Ｅ」を通り過ぎてしまっても、慌てないで4つのカーソル・キーをうまく使って「Ｅ」のところに持って行ってください。

```
10 PL[E]Y"CRCRGRGRARARGGRR"
```

（□はカーソルを表わす）

ここでAを押すと、「E」が「A」に変わってPLAYとなります。カーソルは「Y」のところに来ていますから、そのままそこで⏎キーを押してください。

これで**修正**が完了しました。

試しにプログラムを見てみましょう。

```
LIST⏎
10 PLAY"CRCRGRGRARARGGRR"
Ok
■
```

それでは次に、1行入力中、⏎を押す前に入力の誤りに気づいた場合についてみてみましょう。

```
20 PLEY■
```

ここまで打って、PLAYの「A」が「E」になっている誤りに気づいたとき、それをなおすには2通りの方法があります。

まず、第1の方法は左向き矢印のカーソルコントロールキー←を2回打ってカーソルを「E」のところに持って行き、Aと打つ方法がありますが、第2の方法として INS DEL キーを2回打って、

```
20 PL■
```

「EY」の2文字を消してしまい、A、Yと打って修正する方法もあります。

さて、どちらかの方法で修正が終ったらプログラムの入力を続けることにしましょう。

5.2 効率のよい入力方法

```
20 PLAY"FRFRERERDRDRCCRR" ⏎
30 PLAY"GRGRFRFRERERDDRR" ⏎
```

次の40番の行が上の30番とまったく同じステートメントでよいときは、カーソルを30の「3」のところに持って行って、4を打ち⏎キーを押します。

```
20 PLAY"FRFRERERDRDRCCRR"
40 PLAY"GRGRFRFRERERDDRR"
■
```

これで30番とまったく同じ行がもう1つ、40番の行に作られました。

プログラムを見てみましょう。

```
LIST⏎

10 PLAY"CRCRGRGRARARGGRR"
20 PLAY"FRFRERERDRDRCCRR"
30 PLAY"GRGRFRFRERERDDRR"
40 PLAY"GRGRFRFRERERDDRR"
Ok
■
```

どうです。40番の行がきちんとできているでしょう。

次の50番の行は10番の行とまったく同じでよいのですが、どうすればよいのでしょうか？
そうです。10の「1」のところにカーソルを持って行って5と打って⏎キーを押せばよいのです。

```
50 PLAY"CRCRGRGRARARGGRR"
20 PLAY"FRFRERERDRDRCCRR"
30 PLAY"GRGRFRFRERERDDRR"
40 PLAY"GRGRFRFRERERDDRR"
Ok
■
```

次の60番の行は20番の行とまったく同じです。いまちょうど、カーソルが20の「2」のところに来ていますね。その位置で6と打って⏎キーを押してください。

```
50 PLAY"CRCRGRGRARARGGRR"
60 PLAY"FRFREREROROROCCRR"
30 PLAY"GRGRFRFREREROORR"
40 PLAY"GRGRFRFREREROORR"
Ok
■
```

プログラムの上にカーソルがあり、プログラムを書き換えてしまう恐れのあるときは、[CLR HOME] キーを使って画面を消してしまいましょう。

[SHIFT] キーを押しながら、[CLR HOME] キーを押してみてください。画面がすっかりきれいになりましたね。

ここで、プログラムを見てみましょう。

LIST⏎

```
10 PLAY"CRCRGRGRARARGGRR"
20 PLAY"FRFREREROROROCCRR"
30 PLAY"GRGRFRFREREROORR"
40 PLAY"GRGRFRFREREROORR"
50 PLAY"CRCRGRGRARARGGRR"
60 PLAY"FRFREREROROROCCRR"
Ok
■
```

これは「キラキラ星」でだれもが知っている有名な曲ですが、このままこのプログラムを実行すると、テンポが非常に遅いながらも曲が流れます。

RUN⏎

音楽を途中で止めたいときは、[SHIFT] キーを押しながら、[BREAK] キーを押してください。一般的に、実行中のプログラムを止めるには、[SHIFT] キーを押しながら、[BREAK] キーを押します。

曲のテンポを速くするには、曲のテンポを指定するステートメントを入れます。

```
5 PLAY300 ⏎
```

ついでに、画面を消すステートメントと、タイトルを表示するステートメントも入れましょう。

```
3 CLS ⏎
4 PRINT"ｷﾗｷﾗﾎﾞｼ" ⏎
```

LIST⏎

```
3 CLS
4 PRINT"ｷﾗｷﾗﾎﾞｼ"
5 PLAY300
10 PLAY"CRCRGRGRARARGGRR"
20 PLAY"FRFREREROROROCCRR"
30 PLAY"GRGRFRFREREROORR"
40 PLAY"GRGRFRFREREROORR"
50 PLAY"CRCRGRGRARARGGRR"
60 PLAY"FRFREREROROROCCRR"
Ok
■
```

行番号を揃えて、

RENUM⏎
Ok
■
LIST⏎

```
10 CLS
20 LOCATE12, 5:PRINT"ｷﾗｷﾗﾎﾞｼ"
30 PLAY300
40 PLAY"CRCRGRGRARARGGRR"
50 PLAY"FRFRERERDRDRCCRR"
60 PLAY"GRGRFRFREREROORR"
70 PLAY"GRGRFRFRERERDDRR"
80 PLAY"CRCRGRGRARARGGRR"
90 PLAY"FRFRERERDRDRCCRR"
Ok
■
```

走らせてみると、「キラキラ星」が演奏されます。

5.3 文字の挿入

ここで、タイトルの「キラキラボシ」を画面の中央に表示したいときは、タイトルを書く位置を**LOCATE**というキーワードを使って指定します。

「15 LOCATE12, ␣5⏎」と、15番の1行を入力してもよいのですが、20の番号とPRINTの間に「LOCATE12, ␣5：」を挿入しても同じプログラムになります。

まずカーソルをPRINTの「P」のところに持って行きましょう。そして、SHIFT＋INS DELキーを12回押す（SHIFTを押したままINS DELキーを12回打つ）と、カーソルのあるP以降が1文字ずつ計12文字分右にずれ、挿入するスペースがあきます。

ここで「LOCATE12, ␣5：」と入力して、⏎キーを押すと挿入修正した20番の行が入力されます。

```
20 LOCATE12, 5:PRINT"ｷﾗｷﾗﾎﾞｼ"
30 ............
..................
```

プログラムを確かめてみましょう。

LIST ⏎

```
10 CLS
20 LOCATE12, 5:PRINT"ｷﾗｷﾗﾎﾞｼ"
30 PLAY300
40 PLAY"CRCRGRGRARARGGRR"
50 PLAY"FRFRERERDRDRCCRR"
60 PLAY"GRGRFRFRERERDDRR"
70 PLAY"GRGRFRFRERERDDRR"
80 PLAY"CRCRGRGRARARGGRR"
90 PLAY"FRFRERERDRDRCCRR"
Ok
■
```

さっそく実行してみましょう。画面に「キラキラボシ」と表示されて音楽が流れます。

5.4 文字の削除

LIST[↵]

```
10 CLS
20 LOCATE12, 5:PRINT"ｷﾗｷﾗﾎﾞｼ"
30 PLAY300
40 PLAY"CRCRGRGRARARGGRR"
50 PLAY"FRFRERERDRDRCCRR"
60 PLAY"GRGRFRFRERERDDRR"
70 PLAY"GRGRFRFRERERDDRR"
80 PLAY"CRCRGRGRARARGGRR"
90 PLAY"FRFRERERDRDRCCRR"
Ok
■
```

「LOCATE12, ␣5」を消したいときは、まず、カーソルを消したい文字のすぐ後ろ、ここではPRINTの「P」のところに持って行きます。そして、[INS DEL]キーを12回打つと、カーソルの左隣りの文字が1文字ずつ計12文字消され、それに続く文字の「PRINT…」を引っ張ってきます。ここで[↵]キーを押すと、削除修正した行が入力されます。

```
30 ……
↑
カーソル
```

[SHIFT]キーを押しながら[CLR HOME]キーを押して画面を消してから、プログラムを確かめましょう。

LIST[↵]

```
10 CLS
20 PRINT"ｷﾗｷﾗﾎﾞｼ"
30 PLAY300
40 PLAY"CRCRGRGRARARGGRR"
50 PLAY"FRFRERERDRDRCCRR"
60 PLAY"GRGRFRFRERERDDRR"
70 PLAY"GRGRFRFRERERDDRR"
80 PLAY"CRCRGRGRARARGGRR"
90 PLAY"FRFRERERDRDRCCRR"
Ok
■
```

これまで簡単な例を示しながら、プログラムの編集についてほんの触りの部分を説明しました。

これまでプログラムを作るということが、どのようなことなのか、ある程度おわかりになられたと思います。

❖ プログラム編集方法（その2） ❖

1．カーソルはカーソルキーを使って自由に移動できる。

[↑]カーソルを上へ移動する。

[↓]カーソルを下へ移動する。

[←]カーソルを左へ移動する。

[→]カーソルを右へ移動する。

2．文字を挿入するには

⇨カーソルを削除したい部分のすぐ後に持って行って、[SHIFT]＋[INS DEL]キーを必要なだけ押す。

3．文字を削除するには

⇨カーソルを削除したい部分のすぐ後に持って行って[INS DEL]キーを必要なだけ押す。

4．画面をきれいにするには

⇨[SHIFT]＋[CLR HOME]キーを押す。（＝ＣＬＳ[↵]）

（[SHIFT]＋[CLR HOME]は「[SHIFT]キーを押しながら[CLR HOME]キーを押す」の意味です）

◇ コマンドのまとめ ◇

いままで出てきた基本的なコマンドについてまとめてみましょう。

1．ＣＬＳ[↵]	画面をきれいにする。
2．ＮＥＷ[↵]	記録されたプログラムを消す。
3．ＬＩＳＴ[↵]	記録されているプログラムを画面に表示する。
4．ＲＵＮ[↵]	プログラムを実行する。
5．ＲＥＮＵＭ[↵]	プログラムの行番号をつけなおす。
6．ＤＥＬＥＴＥ　n－m[↵]	プログラムのｎ番からｍ番までを消す。

2章

2章
絵を描く、色を塗る

まず、キーボードより、

INIT ⏎

WIDTH4Ø，25，Ø，1 ⏎

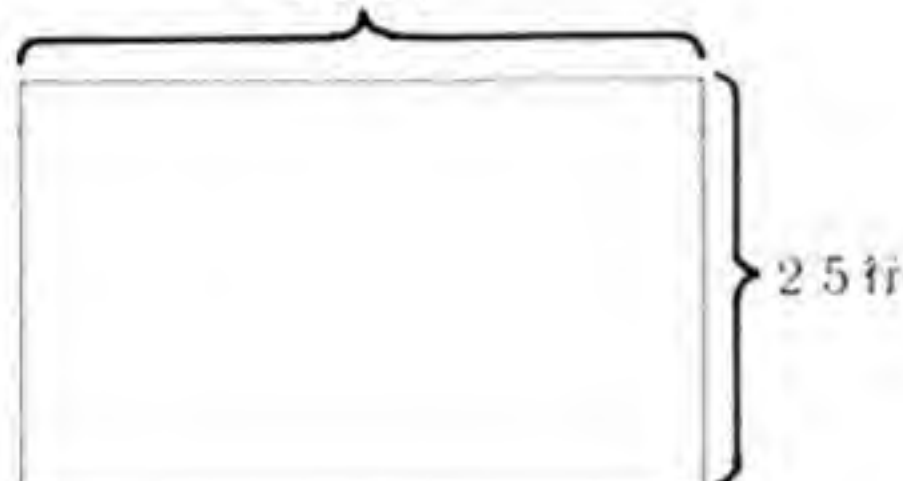
と入力してください。（INIT，WIDTHについては応用編の「ディスプレイモード」で詳しく説明します。）

このように指定すると、

表示文字数　　　　　　4Ø文字×25行（1，ØØØ文字）

グラフィック解像度　　32Ø×2ØØドット

の画面設定になります。

●文字を表示する場合

●図形（グラフィック）を描く場合

本章では、この画面を使用します。

（なお、本機ではWIDTHステートメントにより各種の画面設定ができます。
詳しくは、応用編「ディスプレイモード」を参照してください。）

1 点をうつ

次のように入力してください。

NEW ⏎

```
10 WIDTH 40:INIT      …横4Ø文字×縦25行の設定。
20 SCREEN0,0          …グラフィック画面の使用モードの指定。
30 CLS4               …画面をきれいにします。
40 PSET(160,100)      …白い点を座標（16Ø，1ØØ）に表示。
```

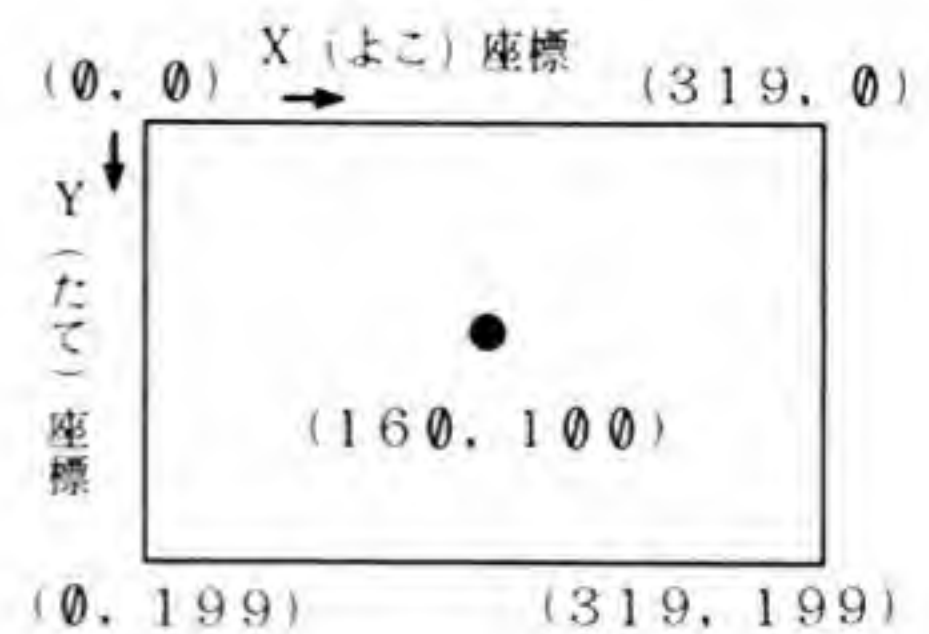
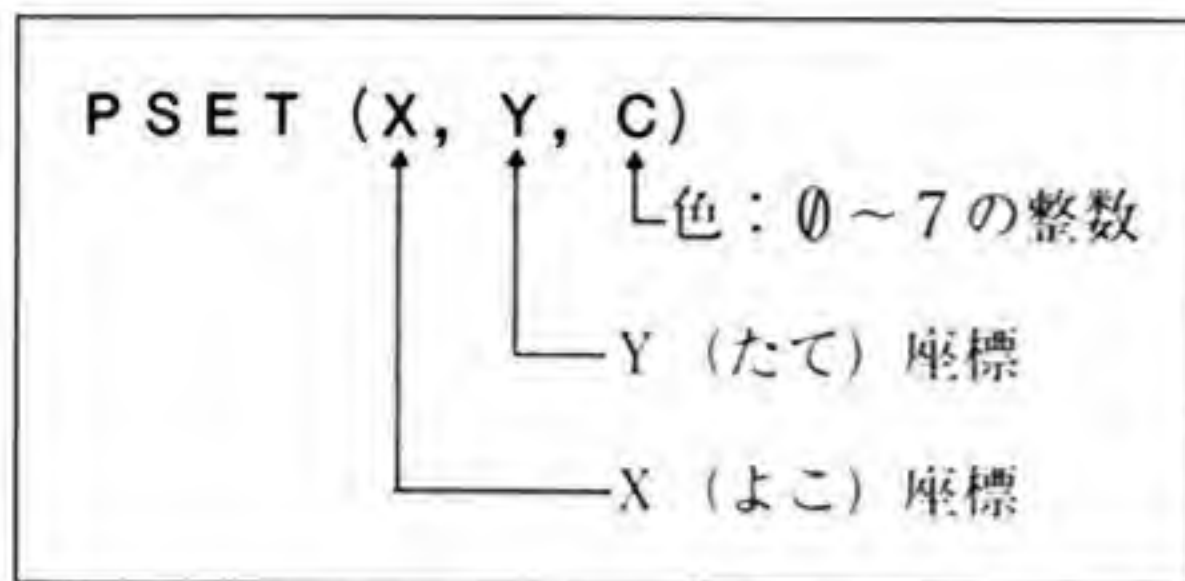
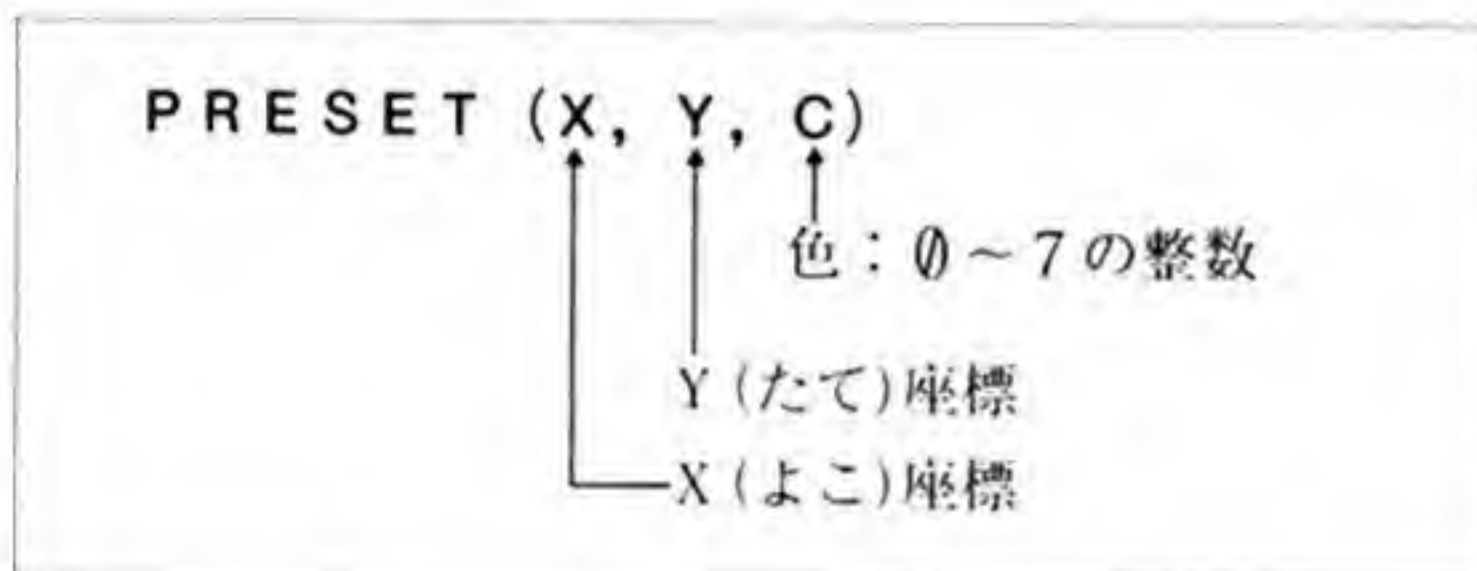
各行ごとに必ず ⏎ キー（リターンキー）を入力してください。そして

RUN ⏎

と入力して実行すると、画面のほぼ中央に白い点が表示されます。

4 0行を

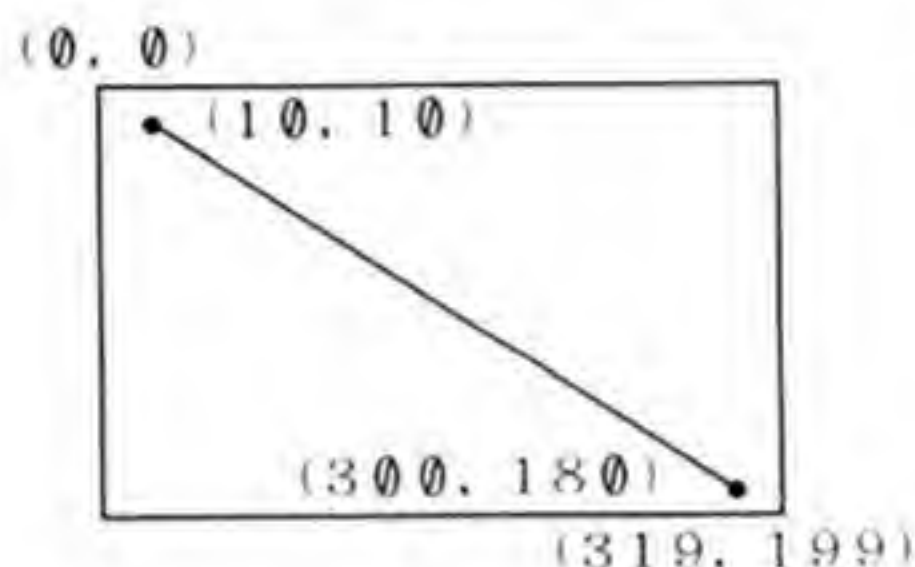
PSET (160, 100, 2)

と変えてみると先ほどと同じ位置に赤色の点が表示されます。

なお、このように、PSETステートメントは点を表示するのに対し、PRESETステートメントは逆に点を消す働きをします。

正確にいえば、PRESETは点を消すのではなく、指定した色を消す命令です。たとえば、白色の点に対してPRESETで緑色を消すようにすると、その点の色はマゼンタに変わります。これは青、赤、緑の3色からなる白色から緑色を消した場合、残るのは青、赤からなるマゼンタだからです。

2　直線を描く

線を描くには、ＬＩＮＥ命令を使います。
たとえば、ＮＥＷ⏎と入力し、

```
10 WIDTH 40:INIT
20 SCREEN0,0
30 CLS4
40 LINE(10,10)-(300,180),PSET,4
```

次にＲＵＮ⏎と実行すると画面の左上から右下へ緑色の直線が描けます。

コンピュータグラフィック花火

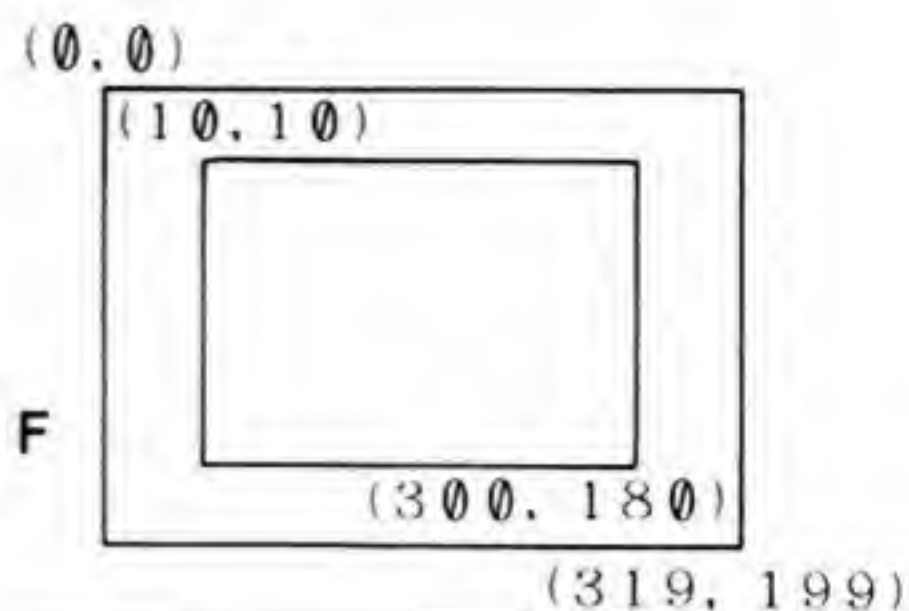
画面上に定めた点と不特定な点とを線で結ぶように描かせるものです。

```
10 REM *** LINE ***
20 WIDTH 40:INIT
30 CLS 4
40 FOR I=1 TO 300
50 X=INT(RND*320)          ← Xに0から319までの整数をランダムに入れる
60 Y=INT(RND*200)          ← Yに0から199までの　　〃
70 C=INT(RND*7)+1          ← Cに1から7までの　　〃
80 LINE(160,100)-(X,Y),PSET,C ← (160, 100)から50, 60, 70行で決められた点に線を描く
90 NEXT
100 END
```

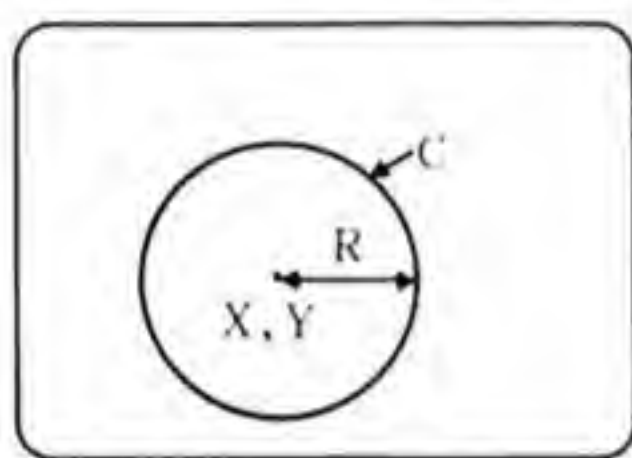
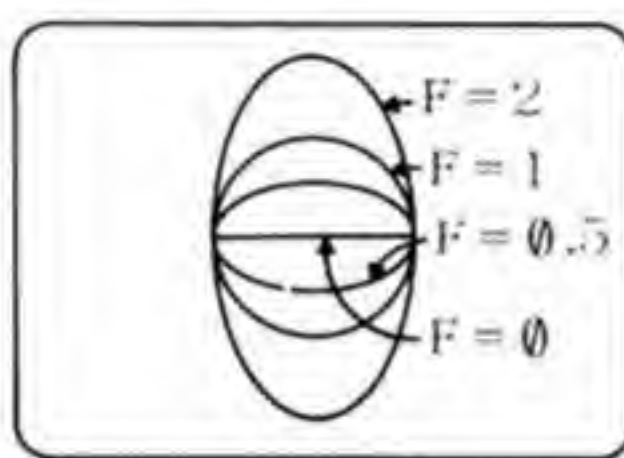
ＬＩＮＥ命令は、他にもいろいろな機能を持っています。詳しくは応用編「グラフィックス」をお読みください。
色の指定の次にＢ（Ｂｏｘ）をつけると、指定した２つの点を対角線とする長方形を描くことができます。たとえば

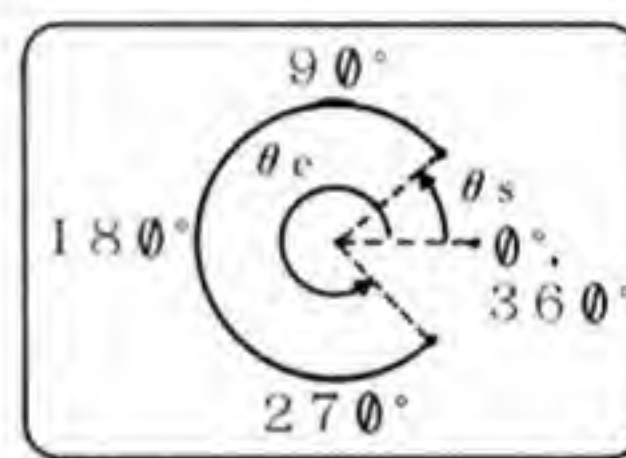
LINE (10, 10) − (300, 180), PSET, 4, B

とすれば、緑色の長方形が描けます。
さらに、BのかわりにBF（Box Fill）とすると、
長方形を描いてその内側を塗りつぶすことができます。

LINE (1Ø, 1Ø) − (3ØØ, 18Ø), PSET, 4, BF

3 円を描く

今度は円を描いてみましょう。円を描くにはCIRCLE命令を使います。

上式でC，F，θs，θeは省略することができます。偏平率Fを1に設定すると、正円になりそれ以外は楕円になります。

円の中心座標　X，Y
半　径　　　　R
円を描かせる色C

偏平率　F
F=2
F=1
F=Ø.5
F=Ø
の参考図

初期角　θs
終了角　θe
θs=45
θe=315
の参考図

WIDTH4Ø：SCREENØ，Ø：CLS4：CIRCLE（16Ø，1ØØ），5Ø，2⏎と入力してください。赤の円が描けますね。
次のサンプルプログラムを入力してください。カラーコードの1から7までを繰り返しながら偏平率と半径を変化させて円を描きます。

サンプルプログラム

カラーサークル

カラーコードの1から7までを繰り返しながら縦横比と半径を変化させて円を描きます。

```
10 REM *** CIRCLE ***
20 WIDTH 40:INIT
30 CLS 4
40 R=1
50 FOR C=1 TO 7
60 CIRCLE(160,100),R,C,N
70 N=N+.05
80 IF R>95 THEN END
90 R=R+5
100 NEXT C
110 GOTO 50
```

4　多角形を描く

次は正多角形を描きましょう。正多角形はＰＯＬＹ命令を使います。
書式としては次のとおりです。

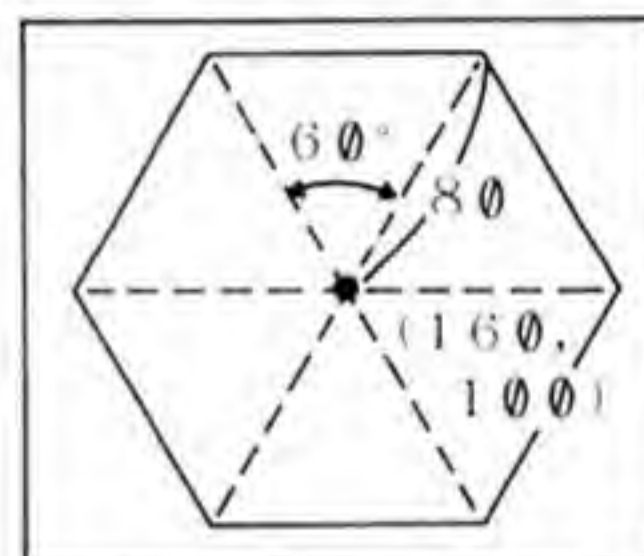
正ｎ角形を描くには次のように設定します。

$\Delta\theta = 360 / n$

$\theta e = \theta s + 360$

たとえば、正六角形を描いてみます。正六角形のステップ角は６０度（３６０÷６）です。

```
10 WIDTH 40:SCREEN0,0:INIT
20 CLS4
30 POLY(160,100),80,2,60
```

これで、中心座標（１６０，１００），頂点距離８０の赤い正六角形が描かれました。

5　色を塗る

線で描いた図形に色を塗ってみましょう。色を塗るにはＰＡＩＮＴ命令を使います。書式としては次のとおりです。

PAINT (X, Y), C, B, B1, B2, …B6

- (X, Y)：塗り始める座標
- C：塗る色
- B, B1, B2, …B6：境界色

ＰＡＩＮＴ命令は（Ｘ，Ｙ）で指定された座標から塗りつぶしを開始し、境界色で指定された色で囲まれた範囲を塗りつぶします。境界色は何色かにまたがる場合がありますので、１色だけでなく何色でも指定できます。
では実際に色を塗ってみます。

```
10 WIDTH 40:INIT
20 CLS4
30 CIRCLE(200,100),80,2
40 PAINT(200,100),1,2
```

これを実行すると、中心座標（２００，１００），半径８０の赤円内を青色で塗りつぶします。

サンプルプログラム

プログラムは、同心円をいくつか描き、円と円で囲まれたドーナツ状の部分を違った色で塗りつぶすものです。

```
10 REM *** PAINT ***
20 WIDTH 40:INIT
30 CLS 4
40 FOR R=1 TO 99 STEP 12 ┐
50 CIRCLE(160,100),R,7   ├←─半径Rが1から99までの同心円を9個描く
60 NEXT R                ┘
70 Y=5
80 FOR C=1 TO 7          ┐
90 PAINT(160,Y),C,7      ├←─同心円の内側をカラーコード1から7までの色で順に塗っていく
100 Y=Y+12               │
110 NEXT C               ┘
```

3章

プログラムの保存と再生

3章

パーソナルコンピュータはプログラムを入力させたり、またそれを実行させたりしますが、これはパーソナルコンピュータの内部にあるＩＣ（集積回路）が処理しています。このＩＣは、いくつものプログラムを入れたり出したりする必要があるため、コンピュータの電源を切ると憶え込んでいたプログラムがすべて消滅するようにつくられています。
ですから、プログラムを保存するには、別の記憶場所へ移さなくてはなりません。
これをプログムのＳＡＶＥ（セーブ）といいます。ここではフロッピーディスクとカセットテープへのＳＡＶＥおよびＬＯＡＤ（再生，ロード）について説明します。
なお、カセットを使用する場合は、オプションの専用データレコーダが必要です。

1 フロッピーディスクに保存、再生する場合

■ＳＡＶＥ……

①プログラムを保存するにはＳＡＶＥ命令を使います。
たとえば、次のプログラムをセーブしようとします。

```
10 INPUT"A=";A
20 INPUT"B=";B
30 C=A+B
40 PRINT"A+B=";C
50 END
■←カーソル
```

②フォーマットされ、ライトプロテクトのかかっていないフロッピーディスクを本体にセットします。

③適当なプログラム名（ただし１３字以内）をつけます。上記のプログラム名を「タシザンノプログラム」として、次のように入力します。

④フロッピーディスクドライブのインジケータが点灯し書き込みを始めます。

⑤画面にＯｋと表示すれば、プログラムはフロッピーディスクにセーブされたことになります。

1)フォーマット …フロッピーディスクを初期設定して、使用可能な状態にすることです。（フォーマットの仕方については『アプリケーションソフトの説明書』の「ディスクユーティリティ」を参照してください。）

2)ライトプロテクト…フロッピーディスクへの書き込み禁止。
ライトプロテクトノッチに保護用紙をはっておくと書き込みはできなくなります。

■LOAD……
①フロッピーディスクから「タシザンノプログラム」を再生するには次のように入力します。

```
LOAD"0:ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ" ⏎
```

再生するプログラムの名前

2 カセットテープに保存、再生する場合

■SAVE……
たとえば、前述の「タシザンノプログラム」というプログラムを保存する例を述べます。
①前述の「タシザンノプログラム」を入力します。
②ツメの折っていないカセットテープを専用データレコーダにセットし、テープカウンターを"000"としておきます。
③

```
SAVE"CAS:ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ" ⏎
```

プログラムを保存するデバイス名（カセット）　保存するプログラムの名前

と入力します。
④専用カセットデータレコーダが自動的に作動を開始します。
⑤画面には
Writing"タシザンノプログラム．"
と表示が出、数秒後にカセットデータレコーダは自動的に停止し、
WRITEの表示ランプは消灯して画面にOkと出ます。
⑥これで「タシザンノプログラム」はカセットテープにセーブできました。

■LOAD?……
以上で、プログラムのセーブは終了したわけですが、万一うまくセーブされてなかったら…
…と考えますね。せっかくのプログラムがセーブされていなければ困りますので
LOAD?という命令で確実にセーブできているか確認します。
①カセットテープをテープカウンター"000"まで巻戻して、セーブした「タシザンノプログラム」が確実にセーブできているか確かめるため次のように入力します。

LOAD?"CAS:ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ" ⏎

確かめるプログラムの名前

②これを実行すると画面には次のようなメッセージが表示され、セーブされていれば数秒後にＯｋが表示されます。

Ｆｏｕｎｄ"タシザンノプログラム．"

Ｏｋ

■

③もし、カセットテープなどに問題があってうまくセーブされていないときは次のメッセージが出ますので、カセットを交換してもう一度セーブをし直してください。

Ｔａｐｅ ｒｅａｄ Ｅｒｒｏｒ

■ＬＯＡＤ……

①プログラムをカセットテープから再生するときは、次のように入力してください。

LOAD"CAS:ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ" ⏎

再生するプログラムの名前

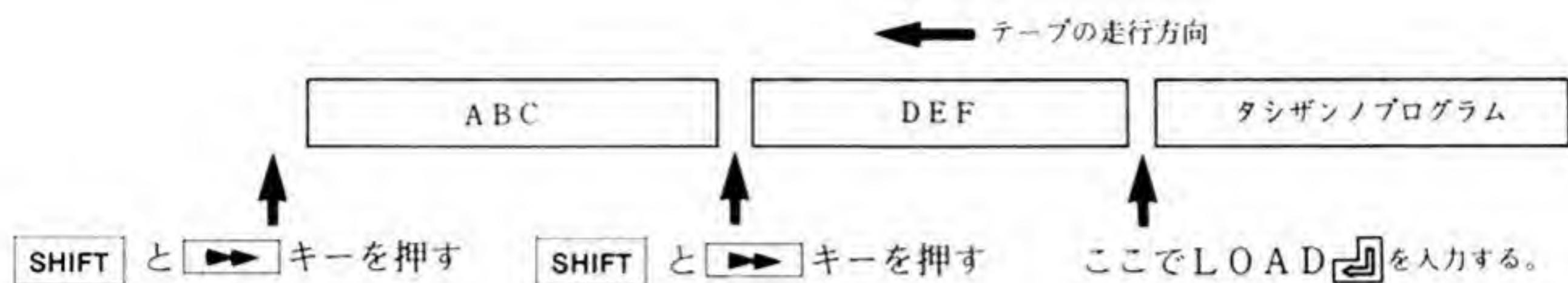
②数秒後、画面に次のようなメッセージが出ます。

Ｆｏｕｎｄ"タシザンノプログラム．"

そしてカセットデータレコーダの動作がとまり、Ｏｋの表示が出て、ロードが完了したことを知らせます。

③なお、１本のカセットテープに何種類かのプログラムが入っていてそのうしろへ先程の「タシザンノプログラム」をセーブした場合はロードするとき最初に説明したように"プログラム名"と一緒に入力します。すると次のようにロードする目的のプログラムに到達するまで関係のないプログラムはスキップしてゆきます。

また、上の例で「タシザンノプログラム」は３番目に入っていることがわかっていれば SHIFT キーを押しながらテープオペレーションキーを押すことによって次のプログラムの頭出しができますからこれを２回繰返すことで「タシザンノプログラム」の頭の部分へすぐに到達することができます。

（次の図を参照してください）

3 フロッピーディスクまたはカセットテープの内容を見るには

プログラムを記録したフロッピーディスクまたはカセットテープの内容を見るときは**FILES**という命令を使います。

この命令を実行すれば、プログラムの名前(ファイル名)だけを拾い出し、その一覧表を画面上に表示します。

■フロッピーディスクの場合

プログラムのはいったフロッピーディスクをディスクドライブ0番にセットし、

```
FILES"0:" ⏎
```

と入力します。

(ディスクBASICを使用したとき、初期状態ではファンクションキー F1 を押せば**FILES"0:"** ⏎を実行します。)

■カセットテープの場合

プログラムのはいったカセットテープをデータレコーダにセットし、テープを巻戻してから

```
FILES"CAS:" ⏎
```

と入力します。

なお、プリンタにファイル名を出力するには、フロッピーディスク、カセットテープそれぞれに対して

```
LFILES"0:" ⏎
```

```
LFILES"CAS:" ⏎
```

と入力します。

4章

テレビをコントロールする

4章

本機では専用ディスプレイテレビを接続したときにキーボードやBASICの命令で自由にテレビをコントロールできます。
さらに本機にはカレンダー付タイマー機能がついていますから、これらを組み合わせて、専用ディスプレイテレビをつけたり、消したりのテレビコントロールが可能になります。ここではあらかじめ内蔵されているテレビタイマーコントロールを使ったものと、BASICの命令を使ったものとの2通りの説明をしています。
ここで使うBASICの命令は次のとおりです。

ASK、TIME$、DATE$、DAY$、TVPW、CRT、CHANNEL、VOL、SCROLL

---

1 テレビタイマーコントロール(TV Timer Contral)

1.1 テレビタイマーコントロールの呼びだし

クロックおよびタイマーの呼び出しは、BASICを読み込んであるかないかによって操作が異なってきます。

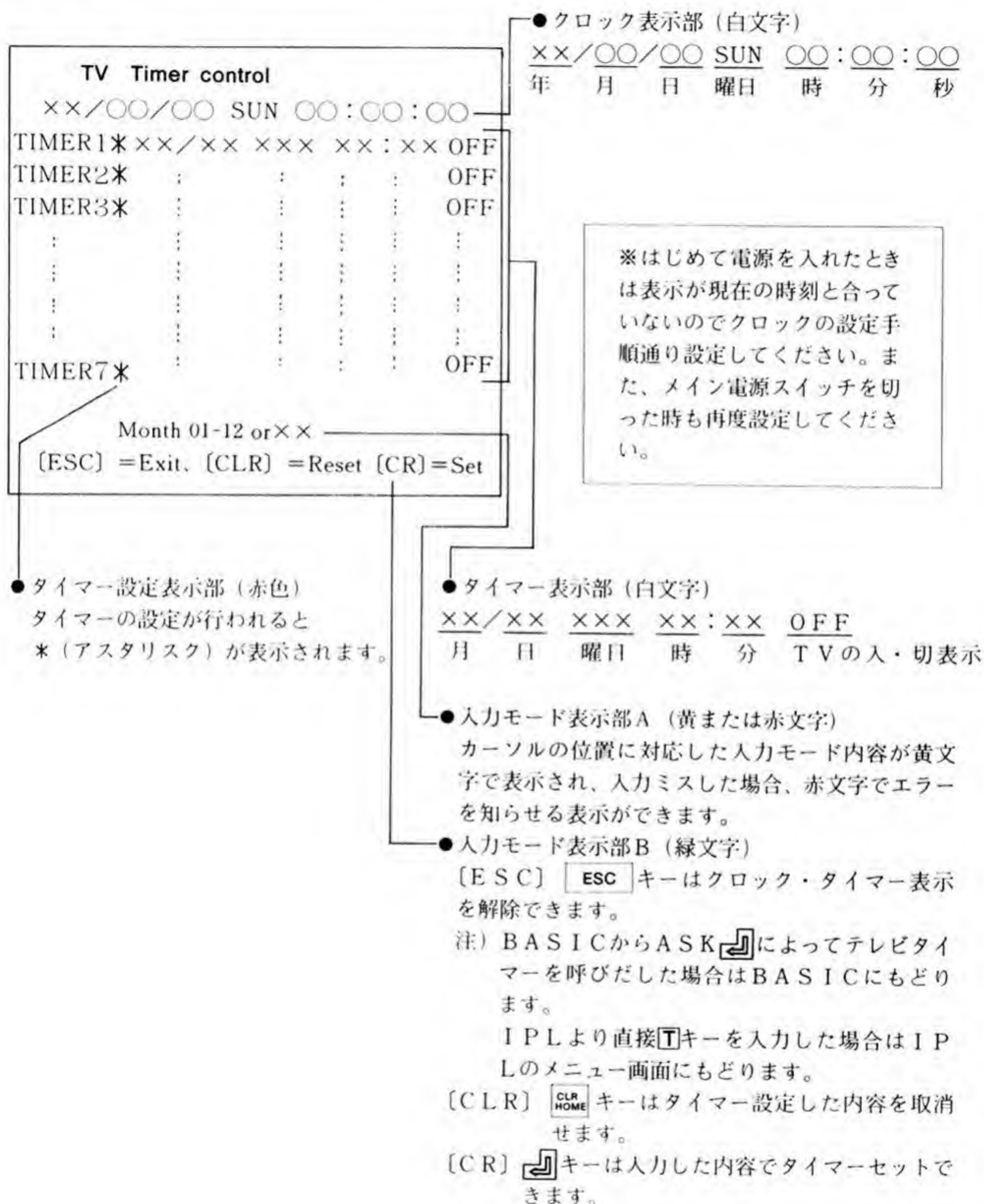
●BASICを読みこんでいない場合

専用ディスプレイテレビの画面上に次のメッセージが表示されますので、Tキーを押してください。

```
Make your device ready
Press selected key to start driving:
      F:Floppy
      R:ROM
      C:CMT
      T:Timer
```

●BASICを読み込ませてある場合はASK↵を入力してください。

1.2 テレビタイマーコントロールの表示内容

画面は下図のようにかわり、カーソルがＴＩＭＥＲ　××／××…の最初の×のところで点滅しています。

1.3 クロック（時計）を現在時刻に合わせる

■現在時刻にコンピュータ内蔵のクロックを合わせる必要があるときは次の手順に従って設定してください。

ここでは１９８６年１月１２日日曜日午後３時３０分２０秒と仮定して説明します。

1. カーソルをカーソルコントロールキーでクロック表示部の左端に移動させます。
2. 西暦１９８６年の末尾２桁［8］［6］をキー入力します。
3. １月の［0］［1］をキー入力します。
4. １２日の［1］［2］をキー入力します。
5. 日曜日の［S］［U］［N］をキー入力します。
6. 午後３時（１５時）の［1］［5］をキー入力します。時刻は２４時間表示ですから午後３時は１５時になります。
7. 午後３時３０分の分の単位［3］［0］をキー入力します。
8. 午後３時３０分２０秒の秒の単位［2］［0］をキー入力します。
9. 設定時刻の３時３０分２０秒になった瞬間に［⏎］キーを押すとその時点からクロックはカウントをはじめ、クロックの設定は終了しカーソルはＴＩＭＥＲ１の頭のところへ移動します。

入力モード表示内容

Year　00-99
Month　01-12　or××
Day　01-31　or××
SUN　MON　THE　WED　THU　FRI　SAT　or×××
Hour　00-23　or××
Minute　00-59
Second　00-59

1.4 タイマーで番組予約をする。

クロックが動作を始めたところで今度は専用ディスプレイテレビの番組予約をするためのタイマー設定を行ってみましょう。

例１　ここでは、先程クロックを設定した翌日つまり１９８６年１月１３日、月曜の午前９時３０分からはじまる５チャンネルの１時間番組を番組予約すると仮定して説明します。

1. カーソルはタイマー表示部のＴＩＭＥＲ１　××…の最初の×のところにきているはずですが、そうでないときは最初の×のところへカーソルコントロールキーで移動します。
2. １月の［0］［1］をキー入力します。
3. １３日の［1］［3］をキー入力します。
4. 月曜日の［M］［O］［N］をキー入力します。
5. 午前９時の［0］［9］をキー入力します。
6. 午前９時３０分の分の単位［3］［0］をキー入力します。
7. カーソルはＯＦＦのＯのところで点滅しています。この例では専用ディスプレイテレビをＯＮさせたいわけですから入力モード表示にこたえるかたちで［Y］をキー入力します。
8. ＯＦＦに変り表示された"ＯＮ　ＣＨ"のうしろでカーソルが点滅していますので予約するチャンネルの［5］をキー入力します。
   最後に［⏎］キー入力するとこの行のタイマーがセットされたことを示す赤色の＊（アスタリス

入力モード表示内容

Month　01-12　or××
Day　01-31　or××
SUN　MON　THE　WED　THU　FRI　SAT　or×××
Hour　00-23　or××
Minute　00-59
TV　Poewr　ON?（Y　or　N）
TV　Channel　1-12

ク）がＴＩＭＥＲ１のあとに表示されカーソルは次の行の頭へ移ります。

ＴＩＭＥＲ１＊１／１３　ＳＵＮ　ØＤ９：３Ø　ＯＮ　ＣＨ５

このようにタイマーセットがなされるとコンピュータ本体インジケータ部の**ＴＩＭＥＲ**表示ランプが点灯し、タイマーが動作中であることを表示します。

1.5 タイマーでスイッチをOFFにする

こんどは専用ディスプレイテレビを指定された時刻にＯＦＦさせるタイマーセットするため、さきほどの手順と同じように

ＴＩＭＥＲ２＊Ø１／１３　ＳＵＮ　１Ø：３Ø　ＯＦＦ

と入力してもよいのですが「１時間後にＯＦＦさせる」のなら次の方法が簡単です。ただし、この場合毎日午前１Ø：３Øには専用ディスプレイテレビをＯＦＦするようにタイマーが働きますので、ご注意ください。

入力モード表示内容

⇩

Month　01-12　or××

TV　Poewr　ON？（Y　or　N）

9．カーソルはＴＩＭＥＲ２　××…の最初の×のところにきています。

10．カーソルキーでカーソルを順次右へ送り、午前１Ø時３Ø分の部分のみ入力します。

ＴＩＭＥＲ２××／××　×××１Ø：３Ø　ＯＦＦ

11．Ｎキーまたは、⏎キーの入力でセットされカーソルは次の行の頭に移ります。

ＴＩＭＥＲ２××／××　×××１Ø：３Ø　ＯＦＦ

これで専用ディスプレイテレビを指定された時刻にタイマーセットできました。

応用例

例２　毎週火曜日の午前１１時から午後２時まで専用ディスプレイテレビをＯＮさせ、１Øチャンネルを番組予約するときのタイマー表示がどうなっているかをごらんください。

ＴＩＭＥＲ３　＊　××／××　ＴＵＥ　１１：ØØ　ＯＮ　ＣＨ１Ø

ＴＩＭＥＲ４　＊　××／××　ＴＵＥ　１４：ØØ　ＯＦＦ

■例１～２の内容を見て、お気付きのように、このタイマーは×（つまり内容を規定しない）を使用することで、プログラマーの思いのままに専用ディスプレイテレビをタイマーコントロールすることができるわけです。

■一度セットされたタイマーの内容は一度メイン電源を切るかまたはクリア（タイマーの取消し）をしない限り、働き続けますので、毎日繰り返すようなタイマー設定の場合は便利です。

1.6 タイマーを取消す

先程セットしたタイマー内容を取消す場合は次の手順で行います。ＴＩＭＥＲ１からＴＩＭＥＲ６を取消すことを例にして説明します。

1．カーソルをＴＩＭＥＲ１の行へ移動させます。この場合カーソルはＴＩＭＥＲ１の行のどの位置でもかまいません。

2．SHIFT キーを押しながら CLR HOME 押すとTIMER１の内容は取消され表示はタイマー設定前の状態にもどります。

3．カーソルはＴＩＭＥＲ２の行に移っていますので同様に SHIFT を押しながら CLR HOME キーを押し

ＴＩＭＥＲ２を取消します。
同じ操作でＴＩＭＥＲ６まで全部取消すとコンピュータ本体の**ＴＩＭＥＲ**表示は消えタイマー設定されていないことを示します。

〈ご注意〉

クロック表示とメイン電源の入・切について
・はじめてメイン電源を入れたときは、表示が現在時刻とちがっていますが、これは故障ではありません。クロックの設定手順を参照して、正確な時刻に合わせてください。
・また、メイン電源を切った状態では、クロック機能が働かなくなりますので、再設定が必要です。
・本機のクロックは、年号の自動切換え表示を行ないませんので、年度が改まるごとに新しい年号を設定してください。
・うるう年（２月２９日のある年）にあたる場合は、月／日の再設定が必要です。
・メイン電源を切りますと年号は××に変わりますので再設定してください。

2 プログラムでテレビをコントロールする

●次のステートメントで内蔵タイマーの設定・表示が行なえます。

```
TIME$="時間：分：秒"
```

```
DATE$="年／月／日"
```

```
DAY$="曜日"
```

次のように入力することで設定・表示が行なわれます。

```
TIME$="12:34:56"⏎
Ok
■
DATE$="85/12/07"⏎
Ok
■
DAY$="FRI"⏎
Ok
■
PRINT  TIME$, DATE$, DAY$⏎
```

```
12:36:03 85/12/07 FRI
Ok
■
```

●BASICでテレビをコントロールする命令はつぎのとおりです。

TVPW	ON	テレビの電源をつけます
	OFF	テレビの電源を消します

CRT	0	テレビ放送を表示します
	1	コンピュータ画面を表示します
	2	テレビ放送のコントラストを下げてコンピュータ画面を重ねて表示します
	3	テレビ放送とコンピュータ画面を重ねて表示します

＊ SHIFT ＋田の状態はCRT2と同じです。

CHANNEL	1～12	チャンネルを設定します

VOL	−62（小さい）～64（大きい）音量を変化させます

SCROLL N	スーパーインポーズ画面のとき(CRT2、CRT3)、コンピュータ画面の上下方向のスクロールを行ないます。Nの値が正のときは上方向に、負のときは下方向にスクロールします。Nの値が0、省略したときストップします。 ただし　N＝−3から3までの整数

ではBASICプログラムでテレビをコントロールしてみます。次のプログラムを入力してみてください。

〔例1〕

```
10 TVPW OFF
20 IF TIME$<>"07:00:00" THEN 20
30 TVPW ON:CHANNEL 1
```

朝7時になるとテレビの電源が入って1チャンネルになります。20行の時刻、30行のチャンネルを変えることで好きな時刻に好きなチャンネルをつけることができます。
これだけではテレビタイマーコントロールで設定するのと同じですから、テレビタイマーコントロールではできない機能を使って少し工夫したプログラムにしてみます。

〔例2〕

```
10 INIT:WIDTH 40,25,0,1:CLS4
20 CRT0:CHANNEL 5
30 CRT 2
40 CSIZE3:LOCATE 0,0:PRINT#0 TIME$
50 GOTO 40
```

20行で指定したチャンネルに時間を大きく表示するようにしたプログラムです。

5章
漢字を表示する

本機には、強力な日本語処理機能を備えた漢字ＢＡＳＩＣ（ＣＺ-８ＦＢ０２）が標準装備されています。この章では、この漢字ＢＡＳＩＣを利用して、漢字、ひらがな等を表示してみます。
（詳細は応用編「日本語処理」を参照してください。）

1 日本語の入力

5章

ＢＡＳＩＣ・ＣＺ-８ＦＢ０２をロードしてください。（すでにＢＡＳＩＣをロードしている場合も、念のためＩＰＬのスイッチを押すなどしてＢＡＳＩＣをロードし直してください。）

日本語は、日本語入力モードに入ってから入力します。次のようにキー入力してください。

（ａ）CTRL ＋ XFER （CTRL キーを押しながら XFER キーを押す）
または
（ｂ）SHIFT ＋ XFER （SHIFT キーを押しながら XFER キーを押す）

日本語入力モードに入ると、表示画面の最下行に変換フィールドが現われます。

なお、日本語入力モードから通常の状態に戻るには、再び前記（ａ）または（ｂ）を実行します。

(1) 漢字の入力

では、実際に日本語を入力してみましょう。
(例)「日本」　　と入力するには…
① N I C H I 　とキー入力します。

② XFER キーを押して漢字に変換します。このとき、"にち"という読みを持つ漢字は、音訓辞書中には"日"しかありませんので、漢字グループを表示せずにテキストエリア上へ漢字"日"を表示します。

③ 次に、H O X をキー入力します。（X キーは「ん」に対応しています。詳しくは後述の「カナ-ローマ字変換一覧表」をご覧ください。）

④ XFER キーを押して漢字変換します。すると、"ほん"の読みを持つ漢字グループがあらわれます。

⑤ 漢字グループの中の該当文字"本"を選びます。⏎キー（リターンキー）を押すか、テンキー 1 を入力します。

以上で、漢字"日本"が入力されました。

次に、HTAB キーを使って、"日本"を表示してみましょう。

① N I C H I とキー入力します。

② HTAB キーを押します。すると記号" ̂I "が表示されます。

③ 続けて H O X とキー入力します。

④ XFER キーを押します。すると、まず最初の"にち"が"日"に変換されます。

⑤ 続けて、XFER キーを押すと、次の"ほん"の読みをもつ漢字グループがあらわれます。

⑥ 漢字グループの中の該当文字"本"を選びます。

以上で漢字"日本"が入力されました。

このように HTAB キーを用いると、1度に漢字数文字分の読みを変換フィールドに入力でき、次々に漢字変換していくことができます。この時、HTAB キーは漢字の読みと読みの間に入力します。後は、XFER キーを押して漢字変換をくり返します。

変換フィールドに入力できるかな数は、全角文字で20字、半角文字で40字です。

HTAB 記号" ^ "は半角文字として入力されますので、1度に変換できる漢字数はこれらより求めることができます。

1）全角文字…漢字やひらがなのように16×16ドットで構成された文字
2）半角文字…8×8ドットまたは16×8ドットで構成された文字

ここで、漢字入力のしかたをまとめます。

①変換フィールド内に出力されたひらがなまたはカタカナは、XFER キーを押すことによってその読みに対応した漢字に変換されます。
漢字は画面右下に最高9文字ずつ系列されます。

②目的の漢字が見つからない場合は、XFER キー、スペースキー、またはカーソルコントロールキー ⇩ を押します。次の漢字グループが表示されます。
また、前のグループに戻すには、カーソルコントロールキー ⇧ を押します。
③目的の漢字が見つかれば、その漢字を取り出します。その方法には次の2通りがあります。
1. 1 ～ 9 のテンキーを使って指定する。
漢字グループの左端の漢字が 1、右端の漢字が 9 のキーに対応していますので、目的の漢字に対応したテンキーを押します。
2. カーソルコントロールキー ⇦、⇨ を使って指定する。
カーソルコントロールキーを使って目的の漢字の上にカーソルを合わせ,リターンキー ↵ を押します。
④以上の操作によって、指定された漢字がテキストエリア内のカーソルの位置に表示されます。
⑤なお、漢字グループを表示した状態で、入力誤りに気づいたり、目的の漢字がなくて変換を中止したい場合は

ESC キー

を押します。すると、漢字グループの表示が消え、変換フィールド内にカーソルが戻りますので、ひらがなまたはカタカナの訂正ができます。なお、変換フィールド内で文字を入力しているとき、間違って入力した場合は

INS DEL キー

を押して入力し直してください。カーソルコントロールキー ⇦ を使用しても働きません。

(2) ひらがなの入力

(例)「ひらがな」と入力するには…
① H I R A G A N A とキー入力します。

② リターンキー (↵) を押せばテキストエリアに"ひらがな"を表示します。

(3) カタカナ、英数字の入力（全角文字）

（例1）「カタカナ」と入力するには…

① ファンクションキー F3 を押すと、〔平仮名〕が〔片仮名〕に変わります。

② K A T A K A N A とキー入力します。

③ リターンキー（⏎）を押せばテキストエリアに"カタカナ"を表示します。

（例2）「ＡＢＣａｂｃ１２３」と入力するには…

①ファンクションキー F4 を押すと、下の画面になります。

キャピタルロックキー CAPS LOCK をロック状態にすると〔英数大〕になり、ロックを解除すると〔英数小〕になります。

②〔英数大〕で A B C とキー入力し、CAPS LOCK キーのロックを解除して
〔英数小〕で A B C とキー入力し、さらにテンキーの 1 2 3 とキー入力します。

③ リターンキー（⏎キー）を押せば、テキストエリアに"ＡＢＣａｂｃ１２３"を表示します。

なお、この状態で次に漢字、ひらがなを表示するには、F 5 キー、続いて F 3 キーを押してください。次の画面になり、漢字、ひらがなが入力できます。

ここで、ファンクションキー F 3 ～ F 5 をまとめます。詳しくは応用編「日本語処理」をお読みください。

F 3 ：平仮名／片仮名…ひらがなまたはカタカナを出力するかを選択します。
F 4 ：英数字直接方式…キーボード上のアルファベット、数字、記号の通りに入力します。
F 5 ：ローマ字-カナ変換方式…アルファベットを入力して、ひらがなまたはカタカナに変換します。

ローマ字-カナ変換一覧表

あ	い	う	え	お				
A	I	U	E	O				
か	き	く	け	こ	きゃ	きゅ		きょ
KA	KI	KU	KE	KO	KYA	KYU		KYO
さ	し	す	せ	そ	しゃ	しゅ	しぇ	しょ
SA	SI	SU	SE	SO	SYA	SYU		SYO
	SHI				SHA	SHU	SHE	SHO
た	ち	つ	て	と	ちゃ	ちゅ	ちぇ	ちょ
TA	TI	TU	TE	TO	TYA	TYU		TYO
	CHI	TSU			CHA	CHU	CHE	CHO
な	に	ぬ	ね	の	にゃ	にゅ		にょ
NA	NI	NU	NE	NO	NYA	NYU		NYO
は	ひ	ふ	へ	ほ	ひゃ	ひゅ		ひょ
HA	HI	HU	HE	HO	HYA	HYU		HYO
		FU						
ま	み	む	め	も	みゃ	みゅ		みょ
MA	MI	MU	ME	MO	MYA	MYU		MYO
や		ゆ		よ				
YA		YU		YO				
ら	り	る	れ	ろ	りゃ	りゅ		りょ
RA	RI	RU	RE	RO	RYA	RYU		RYO
わ		を		ん				
WA		WO		X				
が	ぎ	ぐ	げ	ご	ぎゃ	ぎゅ		ぎょ
GA	GI	GU	GE	GO	GYA	GYU		GYO
ざ	じ	ず	ぜ	ぞ	じゃ	じゅ	じぇ	じょ
ZA	ZI	ZU	ZE	ZO	ZYA	ZYU		ZYO
	JI				JA	JU	JE	JO
だ	ぢ	づ	で	ど	ぢゃ	ぢゅ		ぢょ
DA	DI	DU	DE	DO	DYA	DYU		DYO

ば	び	ぶ	べ	ぼ	びゃ	びゅ	びょ
BA	BI	BU	BE	BO	BYA	BYU	BYO
ぱ	ぴ	ぷ	ぺ	ぽ	ぴゃ	ぴゅ	ぴょ
PA	PI	PU	PE	PO	PYA	PYU	PYO
ふぁ	ふぃ		ふぇ	ふぉ			
FA	FI		FE	FO			

① 小文字は、[SHIFT] キーを押しながらアルファベットを入力することによって変換できます。

「ぁ」……………[SHIFT] + [A]
「ぃ」……………[SHIFT] + [I]
「ぅ」……………[SHIFT] + [U]
「ぇ」……………[SHIFT] + [E]
「ぉ」……………[SHIFT] + [O]
「ゃ」……………[SHIFT] + [Y] [A]
「ゅ」……………[SHIFT] + [Y] [U]
「ょ」……………[SHIFT] + [Y] [O]
「っ」……………[SHIFT] + [Z] または [SHIFT] + [T] [U] または [SHIFT] + [T] [S] [U] または [SHIFT] + [illegible]

小文字の「っ」は、上記入力の他に、同じアルファベットを２度連続して入力することによって表示させることもできます。

〔例〕ローマ字入力「[I]、[R]、[A]、[S]、[S]、[H]、[A]、[I]」→カナ変換出力「いらっしゃい」

② 「ん」は、次の様にも入力できます。

「ん」……………「[SHIFT] + [N]」または「[N]、[SHIFT] + [illegible]」

③ 「きゃ」、「ちぇ」、「ぴゃ」などの文字は、大文字、小文字を別々に入力することもできます。

〔例〕「きゃ」………「[K]、[Y]、[A]」または「[K]、[I]、[SHIFT] + [Y] [A]」

大文字「き」入力　　　小文字「ゃ」入力

〔注〕上記の説明の中で、入力キーをカンマ（,）で区切った場合は、それらのキーを連続して入力することを示します。逆にプラス（+）で接続した場合は、それらのキーを同時に入力することを示します。

例「[N]、[SHIFT] + [illegible]」………[N] キーを入力した後連続して、[SHIFT] キーを押しながら [illegible] キーを入力することを示します。

※特殊文字の入力

入力方式がローマ字-カナ変換方式（[F5]）または [カナ] キーロックによるカナ入力方式の場合、日本語文書作成の上で欠かすことのできない特殊文字を入力することができます。

① 濁点「〃」及び半濁点「°」の入力には、次のキーを押します。

「〃」………[illegible]

「°」………[illegible]

また間接出力モード（ F 2 ）の場合、変換フィールド内でひらがなやカタカナの直後に濁点又は半濁点を入力すると、自動的に濁点、半濁点のついた1文字のカナに変換されます。

〔例〕「し」の後に「゛」を入力すると、自動的に「じ」に変換されます。

逆に直接モード（ F 2 ）を選択した場合、カナと濁点は各1文字分のスペースをとりますので、2文字として扱われます。

② 読点「、」及び句点「。」の入力には、次のキーを押します。

「、」……… SHIFT ＋ [<, ､]

「。」……… SHIFT ＋ [>. ｡]

③ 始めかぎ括弧「「」及び終わりかぎ括弧「」」の入力には、次のキーを押します。

「「」……… SHIFT ＋ [[｢]

「」」……… SHIFT ＋ [ ] ｣]

2 ミニ・ワープロ機能

本機のBASIC（CZ-8FB02）では、漢字変換機能だけでなく簡単なワードプロセッサとしての機能を持っています。この機能を利用すれば、日本語文章を画面に表示させたり、プリンタに印字させたりすることが、簡単に行なえます。本機のBASICでは、そのために、次のコマンド及びステートメントを用意しています。

LIST* ……… 画面表示用コマンド
LLIST* ……… プリンタ印字用コマンド
CONSOLE# ……… 印字エリア設定用ステートメント
AUTO* ……… REMステートメント付き行番号発生用コマンド

(1)LIST*

まず、アポストロフィ「'」（REMステートメントの省略形）を使って次のプログラムを入力してください。

```
10 ' パソコンテレビX1 turbo
20 '   パーソナルコンピュータ
30 '     CZ-856C 新発売
```

ここでLIST* ⏎ を実行すると、次の画面のように、行番号や「'」もないふつうの文章になっています。

```
パソコンテレビX1 turbo
  パーソナルコンピュータ
    CZ-856C 新発売
```

また、ＬＩＳＴ＊の終わりに行番号を指定することによって、任意の行番号の文章だけの表示も可能です。

(2)ＬＬＩＳＴ＊

ＬＬＩＳＴ＊コマンドは、アポストロフィ「'」に書かれている文章だけをプリンタに印字させる働きをします。したがってメッセージや説明文、手紙などの文章をアポストロフィ「'」を使って、プログラム中に作成することが可能となります。

(3)ＣＯＮＳＯＬＥ＃

ＣＯＮＳＯＬＥ＃コマンドを使えばプリンタの印字エリアを設定することができますので、ハガキ大のものから大判サイズのものまで、自由に印字することができます。

(4)ＡＵＴＯ＊

アポストロフィ「'」を使ったプログラムの作成には、ＡＵＴＯ＊コマンドを利用すると便利です。ＡＵＴＯ＊コマンドは、自動的にアポストロフィ「'」の行番号を発生させますので、プログラムということを意識することなく、文章を作成することができます。ただし、１つの行番号に入力できる文字数には限りがありますので注意してください（半角文字の場合２５０文字程度、全角文字の場合１２５文字程度）。

以上のように、本機のＢＡＳＩＣでは多彩な日本語処理機能が追加されていますので、有効に活用すれば簡単なミニ・ワープロとして利用できます。

3　文字のコピー機能：COPY キー

(1)画面上のコピー

本機の漢字ＢＡＳＩＣには、画面上に表示されている文字列（全角文字、半角文字を問わず）を任意の位置にそっくりコピーする機能があります。

コピーの方法を以下に示します。

①まず、コピーしたい位置にカーソルを移動し COPY キーを押します。すると、カーソルの点滅が止まります。

②次に、カーソルコントロールキーでコピーしたい文章の先頭へカーソルを移動します。この時、移動したカーソルは点滅しています。

③⏎キーを押すごとに、目的の位置に１文字ずつコピーされます。

途中でカーソルコントロールキーを押せば点滅している方のカーソルが移動しますので、任意の箇所からコピーを再開することができます。

④コピー先（点滅していない方のカーソル）が行割れをおこす（次の行になること）と自動

あけまして　おめでとう。
↑②コピーしたい文章の先頭に
↑　カーソルをもってくる。
■←コピーしたい位置
①COPYキーを押す

あけまして　おめでとう。

あけまして■
③⏎キーを押すごとに
コピーされる。

的にコピー機能が停止します。

また、コピーを中止したいときは再び COPY キーを押すか、 ESC キーを押してください。

(2)ハードコピー

COPY キーは、前述した画面上のコピー機能だけでなく、プリンタへのハードコピー機能も持っています。ハードコピーできる画面は、テキスト画面のみ、グラフィック画面のみ、テキスト画面とグラフィック画面を合成した画面の3通りです。ハードコピーを行なうためには、次のキーを入力します。

SHIFT ＋ COPY ………テキスト画面のみハードコピー
GRAPH ＋ COPY ………グラフィック画面のみハードコピー
CTRL ＋ COPY ………テキスト画面とグラフィック画面を合成した画面のハードコピー

ただし以下の画面モードではハードコピーできません。

４０文字×２０行、８０文字×２０行、４０文字×１０行、８０文字×１０行

第2部

応用編

1章
プログラムの編集について

ＢＡＳＩＣプログラムを作成するときに、誤って隣りのキーを押してしまったり、途中の文字を抜かしてしまったりする場合があります。また、行やステートメントを新しくつけ加えたり、今まであったものをとってしまったりする編集作業をすることがあります。このような時のために、いろいろな**画面編集（スクリーンエディタ）**機能が用意されています。

1 基本的な編集機能

(1)カーソルの移動

●文字を修正したい時には、その修正箇所までカーソルを移動させます。カーソルを移動させるには以下のカーソルコントロールキーを使用します。

⇨を押すとカーソルは右へ移動します。画面の右端にカーソルがあるときにこのキーを押すと１つ下の行の左端に移動します。また、画面の一番下の行の右端にカーソルがあるときにこのキーを押すと、画面は１行上へスクロール（巻きあがり）し、カーソルは一番下の行の左端に移動します。

⇦を押すとカーソルは左へ移動します。画面の左端にカーソルがあるときにこのキーを押すとカーソルは１つ上の右端に移動します。また、画面の一番上の行の左端にカーソルがあるときにこのキーを押しても、カーソルは画面右はしに移動します。

⇧を押すとカーソルは上へ移動します。画面の一番上の行にカーソルがあるときにこのキーを押してもカーソルは移動しません。

⇩を押すとカーソルは下へ移動します。画面の一番下の行にカーソルがあるときにこのキーを押すと画面は１行上へスクロール（巻きあがり）します。

●通常キーボードのキーを押し続けると同じ文字あるいは同じ動作が継続して入力されます（**リピート機能**）。カーソルコントロールキーも押し続けると押している間カーソルが連続して移動します。ＲＥＰＥＡＴ　ＯＦＦ↵と入力することにより、このリピート機能はなくなり、１文字ずつしか受け付けなくなります。再度リピート機能をはたらかせるにはＲＥＰＥＡＴ　ＯＮ↵と入力します。

●カーソルを瞬時にホーム位置（画面の左上隅）に移動させるには CLR HOME キーを押します。また SHIFT ＋ CLR HOME （ SHIFT キーを押しながら CLR HOME キーを押す）を押すとテキスト画面がすべて消され、カーソルはホーム位置に移動します。ただし、記憶されたプログラムは消えていませんのでＬＩＳＴ↵とすれば画面に表示できます。

(2)文字の削除

文字の削除を行なう場合には、削除したい文字の次の文字にカーソルを移動させた後、 INS DEL キーを押します。この場合、カーソルの左にある文字が削除されカーソルから右の文字列が左へ移動します。

(3)文字の挿入

文字の挿入を行なう場合には、文字を挿入したい位置の次の文字にカーソルを移動させた後 SHIFT + INS/DEL キーを押します。この場合、カーソルから右の文字列が右へ移動し、カーソル位置にはスペース（空白）ができます。このスペース位置に追加したい文字を書き込みます。

このようなスクリーンエディタ機能を用いてプログラムの訂正を行なった場合には、各行番号の文の訂正が終わる度に、必ずキャリッジリターンキー（⏎）を押してください。これにより、画面に表示されいるステートメントの文字列がプログラムとしてメインメモリに登録されます。

（例）「 10 PLIINT CH$(65) 」を「 10 PRINT CHR$(65) 」に訂正する場合次のようにしてください。

■はカーソルを示します。

```
10 PLIINT CH$(65) ■
```

⇦キーを押してカーソルを「L」の位置に移動させ R キーを押します。

```
10 PR[I]INT CH$(65)
```

⇨キーを押してカーソルを次の「I」の位置に移動させ INS/DEL キーを押して「I」を1文字削除します。

```
10 PR[I]NT CH$(65)
```

⇨キーを押してカーソルを「$」の位置に移動させ SHIFT + INS/DEL キーを押して「H」と「$」の間に1文字分のスペースを作ります。

```
10 PRINT CH■$(65)
```

R キーを押して「H」と「$」の間に文字「R」を挿入します。

```
10 PRINT CHR[$](65)
```

⏎キーを押してプログラムをメインメモリへ登録します。

以上のスクリーンエディタ機能を利用してプログラムの修正は行なえますが、画面編集をより効果よく迅速に行なうために、さらに便利な機能として次のような機能が用意されています。

2 知っていると便利な編集機能

(1)カーソルの移動

●水平ＴＡＢ

画面の初期状態では水平ＴＡＢ位置は画面左端から８文字単位に設定されています。H TAB キーを押すとカーソルは現在位置から次の水平ＴＡＢ位置（初期状態では８文字右）へ瞬時に移動します。この機能を利用すると、文字の始まり位置を統一することが容易に行なえます。また、水平ＴＡＢ位置は表示画面内で任意に設定できます。まず、設定されているＴＡＢ位置を取り消す場合にはその取り消す水平ＴＡＢ位置へカーソルを移動させ CTRL + Y キーを押します。逆に新しく水平ＴＡＢを設定する場合には、設定したい水平位置にカーソルを移動させ CTRL + T キーを押します。

●１ワード単位でのカーソル移動

CTRL + F キーを押すとカーソルは現在位置より右（先）にあるワードの先頭に移動します。また、CTRL + B キーを押すとカーソルは現在位置より左（後）にあるワードの先頭に移動します。ここで、ワードとは空白、記号を除く連続した文字列のことです。ただし、コロン（:）はワードとみなされますが、πはワードとはみなされません。

(2)文字の削除

●カーソル位置からその行の終わりまでの削除

CTRL ＋ E キーを押すとカーソル位置からその行の終わりまでの文字が削除されます。この後で、キャリッジリターンキーを押すことによりプログラムが修正されます。

●カーソルのある位置以降の画面クリア

CTRL ＋ Z キーを押すとカーソル位置以降の画面をすべてクリアします。

(3)文字の挿入

文の途中で何文字か挿入したい場合には、文字を挿入したい位置の次の文字にカーソルを移動させ CTRL ＋ A キーを押します。これで、インサートモードに設定され、それ以後に入力された文字がカーソル位置に表示されると共にカーソルより右の文字列が右へ移動していきます。インサートモードを解除するには CTRL ＋ S キー（ＢＲＥＡＫと同じ）を押します。また、キャリッジリターンキーを押してもインサートモードは解除されます。また、インサートモード時に INS DEL キーを押して文字を削除してもインサートモードは解除されません。

(4)行の分割

１行の内容を２行にわるように書き直したい場合には、区切りたい箇所へカーソルを移動させ CTRL ＋ J キーを押すと、カーソルからその行の終わりまでの文が、自動的に次の行へ移動します。移動した行の先頭に行番号をつけてキャリッジリターンキーを押せばこれまでの１行の内容が２行にわたって書かれます。

(例) 次に示す行番号５０で定義されているプログラムをＣＩＲＣＬＥ文とＰＡＩＮＴ文に分離してみましょう。

```
50 CIRCLE(100,100),50,1:PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

カーソルをＰＡＩＮＴ文の「Ｐ」の位置に移動させ CTRL ＋ J キーを押します。するとＰＡＩＮＴ文以降が次の行へ移動します。

```
50 CIRCLE(100,100),50,1:
PAINT (100,100),HEXCHR$("AA00AA00AA00"),1
```

SHIFT ＋ INS DEL キーを押してＰＡＩＮＴ文の前に行番号を挿入するためのスペースを作り、行番号６０を入力します。

```
60 PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

キャリッジリターンキー↵を押して行番号５０，６０の文をプログラムとしてメインメモリへ登録します。

(5)行の結合

２行にわたる文を１行に納めたい場合には、行の結合機能を利用します。CTRL ＋ W キーを押すと、カーソルのある水平１行と次の１行とが結合されます。よって２行にわたる文が１行として扱われ、これまでの２行の文と文の間隔は INS DEL キーを押して文字を削除することによりつめることができます。結合する２つの行のうち前の１行の文が画面の数行にわたるときには、カーソルをその文の１行内の最下行に移動させて CTRL ＋ W キーを押してください。

(例) 次に示す行番号５０と６０で定義されているプログラムを１つの行番号の文にまとめてみましょう。

```
[5]0 CIRCLE(100,100),50,1:
 60 PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

行番号５０のライン上にカーソルを移動させ CTRL ＋ W キーを押します。これにより行番号５０と６０の文が結合されました。この時点ではまだ、メインメモリへは登録されていません。

```
50 CIRCLE(100,100),50,1:
60[P]AINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

カーソルをＰＡＩＮＴ文の「Ｐ」の位置に移動させ INS DEL キーを押すとＰＡＩＮＴ文が左へ移動し画面左端まで移動します。さらに押し続けると１行上の行番号５０行の画面右端よりＰＡＩＮＴ文が現われます。ＣＩＲＣＬＥ文の右にＰＡＩＮＴ文を並べた時点で ⏎ キーを押します。

```
50 CIRCLE(100,100),50,1:PAINT(100,100),HEXCHR$("AA00AA00AA00"),1■
```

これにより、文番号５０にＣＩＲＣＬＥ文とＰＡＩＮＴ文がプログラムとしてメインメモリへ登録されました。しかし文番号６０にはまだＰＡＩＮＴ文が残った状態となっていますので文番号６０の文を消す必要があります。

```
50 CIRCLE(100,100),50,1:PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
60 ⏎
```

文番号６０のみを入力してキャリッジリターンキーを押すことにより文番号６０のプログラムがメインメモリから削除されます。以上により２つの行番号で定義されていたプログラムが１つの文番号内に定義されました。

(6)文字のコピー

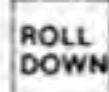
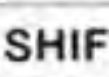
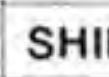
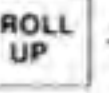
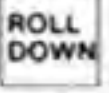
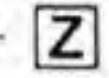
本機は画面に表示されている文字をそのまま別の場所にコピーして表示する機能を持っています。コピーをするにはまず、コピーする文字を表示したい場所にカーソルを移動し、 COPY キーを押します。カーソルは点滅を停止し静止した状態になります。ここで、カーソルコントロールキーを押すと、静止したカーソル位置から点滅したカーソルが現われます。このカーソルをコピーしたい文字の先頭まで移動させます。 ⏎ キーを押すと、点滅したカーソル位置にある文字が点滅を停止したカーソル位置へコピーされ、両カーソルは右へ移動します。コピーを終了するには再度 COPY キーを押します。

このコピー機能によりコピー表示する行は水平１行にのみ有効です。次の行にわたる必要がある場合には以上のコピー操作を再度行なってください。

また、この COPY キーは次のようにプリンタへの画面コピー機能を持っています。

SHIFT ＋ COPY **キーを押すとテキスト画面のみをプリンタにコピーします。**

GRAPH ＋ COPY **キーを押すとグラフィック画面のみをプリンタにコピーします。**

CTRL ＋ COPY **キーを押すとテキスト画面とグラフィック画面をプリンタにコピーします。**

ただし以下の画面モードの時には画面コピーはできません。

４０文字×２０行、８０文字×２０行、４０文字×１０行、８０文字×１０行

(7)行間への新しい行の追加

表示画面の行と行の間をひろげ新しく行を追加したい場合には、 ROLL UP キーあるいは ROLL DOWN キーを使用します。 ROLL UP キーを押すとカーソルのある行から上の行が１行上にスクロールします。また、 ROLL DOWN キーを押すとカーソルのある行から下の行が１行下にスクロールします。これにより、行と行の間がひろげられ新しい行を追加できます。ただし、次の項(8)に示すように、エディットモードではこれらのキーの働きが異なります。

(8)ＥＤＩＴ機能

プログラムを実行させた場合、プログラムに何らかのエラーがあると、エラーの種類とそのエラーが生じた行番号を表示して実行が中断されます。このような場合ＥＤＩＴ文を使用することによりエラーの発生したプログラムの修正を能率よく行えます。例えばＥＤＩＴ ⏎ と入力すると、エラーの発生した行を画面に表示して、カーソルはその先頭に表示され、エディットモードに入ります。エディットモードでは ROLL UP **キー**あるいは ROLL DOWN **キー**を使用することにより、巻き物を見るように、画面に表示されている文の前後を表示できます。この機能を利用して、訂正する文を画面に表示させ訂正します。エディットモードを解除するには SHIFT ＋ CLR HOME キー あるいは SHIFT ＋ BREAK キーを押してください。

エディットモードで ROLL UP キーまたは ROLL DOWN キーを使用して、プログラムを画面全体に表示してスクロールしている状態で、新しく行を追加するには CTRL ＋ E あるいは CTRL ＋ Z **キー**を使用して画面に空白部分を作成してから行番号付文をキー入力し、最後にキャリッジリターンキーを入力してください。

また、ＥＤＩＴ機能はエラーが発生した時のみでなく、プログラムの作成後、プログラムの見直しを行う場合に便利です。ＬＩＳＴ文の実行でもプログラムの見直しはできますが、画面上部へ消えていった行を表示するには再度ＬＩＳＴ文を実行する必要があります。

ＥＤＩＴ文は次の形式で表わします。

```
EDIT  {n}
      {.}
```

n…行番号

.…ピリオド（.）にはエラーが発生したときの行番号が入ります。

(9)行番号の整理

プログラムを修正して行を取り除いたり、新しい行を追加すると行番号がばらばらになります。このようなとき、ＲＥＮＵＭ文を使用して行番号を整理することができます。ＲＥＮＵＭ文は次の形式で表わし、旧行番号 m で指定した行以降の行番号を新行番号 l で始まる行番号に増分 n で付け変えます。この場合、プログラム内のＧＯＴＯ文などでジャンプする行番号も新しい行番号に書き変わります。

```
RENUM  [[l],[m],[n]]
```

l…新行番号。省略すると１０。

m…旧行番号。省略するとプログラムの最初の行。

n…増分。省略すると１０。

3　全角文字の編集

本機は一般の英数カナ文字に加えて、日本語文字も簡単にテキスト画面に表示することができ、プログラム中でも日本語文字を使用できます。ただし、標準解像度モード（縦２００ライン表示モニター用）でテキスト画面が４０×２５、４０×２０、８０×２５、８０×２０行モード時には日

本語文字は表示できません。一般の英数カナ文字1バイトコードで表わされて**半角文字**と呼ばれるのに対して、日本語文字は2バイトコードで表わされ**全角文字**と呼ばれ、文字の大きさも英数カナ文字の水平方向が2倍の大きさで表示されます。

プログラムを編集する際に、全角文字は半角文字と取り扱い方やカーソルの動作が異なります。以下に、全角文字の編集方法について説明します。

(1)全角文字の表示

プログラムの作成中に全角文字を使用する場合には、まず CTRL + XFER **キー**を押して日本語入力モードにします。画面の最下行に変換フィールドが表われますので、表示したい全角文字を指定してキャリッジリターンキーを押すとカーソルの位置に指定した全角文字が2桁（1桁は半角文字1文字分を示します）の大きさで表示されます。再度 CTRL + XFER **キー**を押すと日本語入力モードが解除されます。日本語入力モードに関しては「5章．日本語処理」の項を参照してください。

(2)全角文字上のカーソルの移動

カーソルコントロールキー ⇨ を操作してカーソルを全角文字上に移動するとカーソルは全角文字全体を覆う大きさに変わります。これは全角文字の1桁目にカーソル位置があることを示します。もう1回 ⇨ キーを押すとその全角文字の右半分のみの大きさに変わります。これは全角文字の2桁目にカーソル位置があることを示します。このように、全角文字列の中では ⇨ キーを2回押すことによりカーソルは1文字進みます。カーソルを ⇦ キーにより戻す場合にはカーソルはこの逆の動作を行ないます。

(3)全角文字の削除

全角文字の削除を行なう場合には、削除したい全角文字の次の文字にカーソルを移動（次の文字が全角文字の場合にはカーソル位置はその文字の前半でも後半でもかまいません）させた後、INS/DEL キーを押します。この場合、カーソルの左にある全角文字が削除されカーソルから右の文字列が左へ1文字分（2桁）移動します。

(4)全角文字の挿入

文字列の途中に全角文字を挿入する場合には、挿入したい位置の次の文字にカーソルを移動（次の文字が全角文字の場合には、カーソル位置はその文字の前半でも後半でもかまいません）させた後、SHIFT + INS/DEL **キー**を2回押します。この場合、カーソルから右の文字列が右へ2桁移動し、全角文字1文字分のスペースができます。このスペースに追加したい全角文字を書き込みます。

(5)全角文字の修正

入れまちがえた全角文字を正しい全角文字に置き換えたい場合には、まちがった全角文字の前半の位置にカーソルを移動させた後、正しい全角文字を入力します。もし、カーソルが全角文字の後半にあった場合に新しく全角文字を入力すると、修正したい全角文字の次にある文字を消してしまいます。

このような機能を利用してプログラムの訂正を行った場合には各行番号の文の訂正が終わる度に、必ずキャリッジリターンキー（⏎）を押してください。これにより、画面に表示されている

ステートメントの文字列がプログラムとしてメインメモリに登録されます。

(例)「10 PRINT"本日語の文を使います。"」を

「10 PRINT"日本語文字を使います。"」に訂正する場合次のようにしてください。

■はカーソル位置を示します。

10 PRINT"本日語の文を使います。"■

[←]キーでカーソルを「本」の位置まで移動します。

10 PRINT"[本]日語の文を使います。"

[COPY]キーを押した後、点滅するカーソルを「日」の位置に移動させて[↵]キーを押して「日」を「本」の位置にコピーした後、再度[COPY]キーを押してコピーモードを解除します。

10 PRINT"日[日]語の文を使います。"

[SHIFT]+[XFER]キーを押して日本語入力モードにした後、全角文字、ローマ字入力、音訓変換モードになっていることを確認して[H]、[O]、[X]、[XFER]と入力して「本」を指定して、[↵]キーを押します。

10 PRINT"日本[語]の文を使います。"

[→]キーを4回押してカーソルを「文」の位置に移動させます。

10 PRINT"日本語の[文]を使います。"

[INS DEL]キーを1回押して「の」を削除します。

10 PRINT"日本語[文]を使います。"

[→]キーを2回押してカーソルを「を」の位置に移動させます。

10 PRINT"日本語文[を]使います。"

[SHIFT]+[INS DEL]キーを2回押して「文」と「を」の間に全角文字1字分のスペースを作ります。

10 PRINT"日本語文　を使います。"

日本語入力モードの状態で[J]、[I]、[XFER]と入力して「字」を指定して[↵]キーを押します。[SHIFT]+[XFER]キーを押して日本語入力モードを解除します。

10 PRINT"日本語文字[を]使います。"

[↵]キーを押して修正したプログラムをメインメモリへ登録します。

4 コントロールコード表

以下にコントロールコードの一覧表を示します。なお下の表で CTRL ＋Aは CTRL キーを押しながら A キーを押すことを意味します。

CTRL＋	コード	処 理 内 容
@	00	ダミー。
Aまたはa	01	インサートモード[1]にする。解除は BREAK または ↵ で行なえる。
Bまたはb	02	現在のワード[2]の先頭にカーソルを戻す。
Cまたはc	03	実行を停止する。(＝ SHIFT ＋ BREAK)
Dまたはd	04	画面などを標準の状態に戻す。(＝INIT)
Eまたはe	05	現在のカーソル以降1行を消す。
Fまたはf	06	次のワードの先頭にカーソルを移す。
Gまたはg	07	ビープ音を鳴らす。(＝BEEP)
Hまたはh	08	1文字分消して戻る。(＝ INS DEL)
Iまたはi	09	水平タビュレーションを行なう。(＝ H TAB)
Jまたはj	0A	現在のカーソル以降を次の行に分ける。
Kまたはk	0B	カーソルを画面のホーム位置に移す。(＝ CLR HOME)
Lまたはl	0C	テキスト画面を消去する。(＝ SHIFT ＋ CLR HOME)
Mまたはm	0D	キャリッジターンをする。(＝ ↵)
Nまたはn	0E	現在のカーソルから上を上方向にスクロールする。(＝ ROLL UP)
Oまたはo	0F	現在のカーソルから下を下方向にスクロールする。(＝ ROLL DOWN)
Pまたはp	10	ダミー。
Qまたはq	11	実行の一時停止を解除する。
Rまたはr	12	空白を挿入する。(＝ SHIFT ＋ INS DEL)
Sまたはs	13	実行を一時停止する。[3] (＝ BREAK)
Tまたはt	14	水平タビュレーション位置の新たな設定を行なう。
Uまたはu	15	ダミー。
Vまたはv	16	ダミー。
Wまたはw	17	現在のカーソルがある行と次の行をつないで1つにする。
Xまたはx	18	ダミー。
Yまたはy	19	現在のカーソル位置の水平タビュレーションの解除を行なう。
Zまたはz	1A	現在のカーソルより下のテキスト画面をすべて消去する。
[	1B	ダミー。
¥	1C	カーソルを右へ移動する。(＝→)
]	1D	カーソルを左へ移動する。(＝←)
^	1E	カーソルを上へ移動する。(＝↑)
－	1F	カーソルを下へ移動する。(＝↓)
0		画面の背景色を黒(透明)にする。(＝COLOR,0)
1		画面の背景色を青にする。 (＝COLOR,1)
2		画面の背景色を赤にする。 (＝COLOR,2)
3		画面の背景色をマゼンタにする。(＝COLOR,3)
4		画面の背景色を緑にする。 (＝COLOR,4)
5		画面の背景色をシアンにする。 (＝COLOR,5)
6		画面の背景色を黄色にする。 (＝COLOR,6)
7		画面の背景色を白にする。 (＝COLOR,7)

テンキーの　0		文字グラフィックの色を黒(透明)にする。(＝ＣＯＬＯＲ0)
1		文字グラフィックの色を青にする。　(＝ＣＯＬＯＲ1)
2		文字グラフィックの色を赤にする。　(＝ＣＯＬＯＲ2)
3		文字グラフィックの色をマゼンタにする。(＝ＣＯＬＯＲ3)
4		文字グラフィックの色を緑にする。　(＝ＣＯＬＯＲ4)
5		文字グラフィックの色をシアンにする。(＝ＣＯＬＯＲ5)
6		文字グラフィックの色を黄色にする。　(＝ＣＯＬＯＲ6)
7		文字グラフィックの色を白にする。　(＝ＣＯＬＯＲ7)
／		キャラクタゼネレータのＲＯＭ/ＲＡＭを切り換える。(＝ＣＧＥＮ)
＊		文字を点滅モードとノーマルモードとに切り換える。(＝ＣＦＬＡＳＨ)
－		文字を反転モードとノーマルモードとに切り換える。(＝ＣＲＥＶ)

1）インサートモード（insert mode）挿入したい文字キーを押すたびに、カーソルから右の部分が自動的に右に移動して、キーボードから入力した文字が挿入されるモード。

2）ワード（word）隙間のない英数字の文字列をいいます。

3）プログラムの実行中は、キーを押している間だけ停止し、キーを離すと再び実行を開始します。

2章
ディスプレイモード

ＢＡＳＩＣは、図形文字（アルファベット、数字、漢字、カタカナ、ひらがな、セミグラフィク文字など）、点、線、その他複雑な図形をディスプレイテレビに表示することができます。

アルファベットなどの図形文字を表示する画面を**テキスト画面**、ドットによる図形を表示する画面を**グラフィック画面**と呼び、ディスプレイテレビの画面は、この２種類の画面を重ねて表示していると考えることができます。

1　テキスト画面

テキスト画面に表示できる文字数は､使用するディスプレイが標準ディスプレイか､高解像度ディスプレイかにより異なります。

○標準ディスプレイ（文字数／行）×（行数／画面）

　　４０文字×２５行，８０文字×２５行
　　４０文字×１２行，８０文字×１２行
　　４０文字×２０行，８０文字×２０行
　　４０文字×１０行，８０文字×１０行

○高解像度ディスプレイ

　　４０文字×２５行，８０文字×２５行
　　４０文字×１２行，８０文字×１２行
　　４０文字×２０行，８０文字×２０行

ディスクＢＡＳＩＣを起動した直後は

　標準ディスプレイ使用時には　　８０文字×１２行
　高解像度ディスプレイ使用時には　８０文字×２５行

に設定されています。

テキスト画面の表示文字数は、ＷＩＤＴＨステートメントを使います。

ＷＩＤＴＨ〔１行当たりの文字数〕,〔画面に表示する行数〕

たとえば、４０文字×２５行の表示画面に変更するには

　　ＷＩＤＴＨ　４０，２５ ⏎

と入力します。すると画面をクリアした後、４０文字×２５行の画面に変わります。ＷＩＤＴＨステートメントを使って、次の８通りの設定ができます。

ステートメント	表示文字数	備　　　考
WIDTH40, 10, g, d	40×10	標準ディスプレイモードのときのみ設定可能。 アンダーラインが引ける。 グラフィック画面表示不可。
WIDTH80, 10, g, d	80×10	
WIDTH40, 12, g, d	40×12	
WIDTH80, 12, g, d	80×12	
WIDTH40, 20, g, d	40×20	アンダーラインが引ける。 グラフィック画面表示不可。
WIDTH80, 20, g, d	80×20	
WIDTH40, 25, g, d	40×25	
WIDTH80, 25, g, d	80×25	

※　g：グラフィック画面の解像度の設定

0＝２００／１９２ドット　1＝４００／３８４ドット

d：ディスプレイモードの設定

0＝本体の標準／高解像度切換スイッチの状態に従う。

■：標準ディスプレイモード（ＳＴＡＮＤＡＲＤ）

▬：高解像度ディスプレイモード（ＨＩＧＨ）

1＝標準ディスプレイモード

２＝高解像度ディスプレイモード

(注)　・標準ディスプレイでは、標準ディスプレイモードのみ、高解像度ディスプレイでは、高解像度ディスプレイモードのみ使用可能です。

・専用ディスプレイテレビ(ＣＺ-８５５ＤまたはＣＺ-８５０Ｄ)では、標準ディスプレイモードおよび高解像度ディスプレイモードのどちらでも使用可能です。
その切換えは、ＷＩＤＴＨステートメントの第4パラメータdによって行ないます。

2　グラフィック画面

(1)　横方向の解像度

ＷＩＤＴＨステートメントの第1パラメータによって決まります。

ＷＩＤＴＨ　４０　ならば　３２０ドット

ＷＩＤＴＨ　８０　ならば　６４０ドット

(2)　たて方向の解像度

本機では、グラフィック画面は、２つのグラフィックメモリ（４８ＫＢが２つ）から構成されています。本機の専用ディスプレイテレビ（ＣＺ-８５５ＤまたはＣＺ-８５０Ｄ）は

標準ディスプレイモード　　（たて２００ドット）

高解像度ディスプレイモード　（たて４００ドット）

の切換えができます。

これは、ＷＩＤＴＨステートメントの第３、４パラメータで指定します。

指定の方法は次のとおりです。

たてのドット数	第3パラメータの値	第4パラメータの値
200 400	0 1	1 2

ただし、たて400ドットを指定する場合は、あらかじめ

OPTION SCREEN 0

を実行してください。

※本機のグラフィックメモリは、本来のグラフィック表示以外に、データやプログラムを記憶する普通のメモリとしても使用できます。これを指定するのがOPTION SCREENステートメントです。詳しくは「6章 フリーエリアについて」を参照してください。

OPTION SCREEN0…全部グラフィック用に使用。たて400ライン可

OPTION SCREEN1…半分を変数エリア用に転用。たて200ライン可 (BASIC起動時)

OPTION SCREEN2…変数エリアと外部記憶用に使用。

OPTION SCREEN3…半分を外部記憶用に転用。たて200ライン可

OPTION SCREEN4…全部外部記憶用に使用。

なお、第4パラメータを0にして、本機前面ドア内の標準／高解像度切換スイッチによって

標準ディスプレイモード（STANDARD）▬

高解像度ディスプレイモード（HIGH）　■

に切りかえることができます。

以上を次表にまとめます。

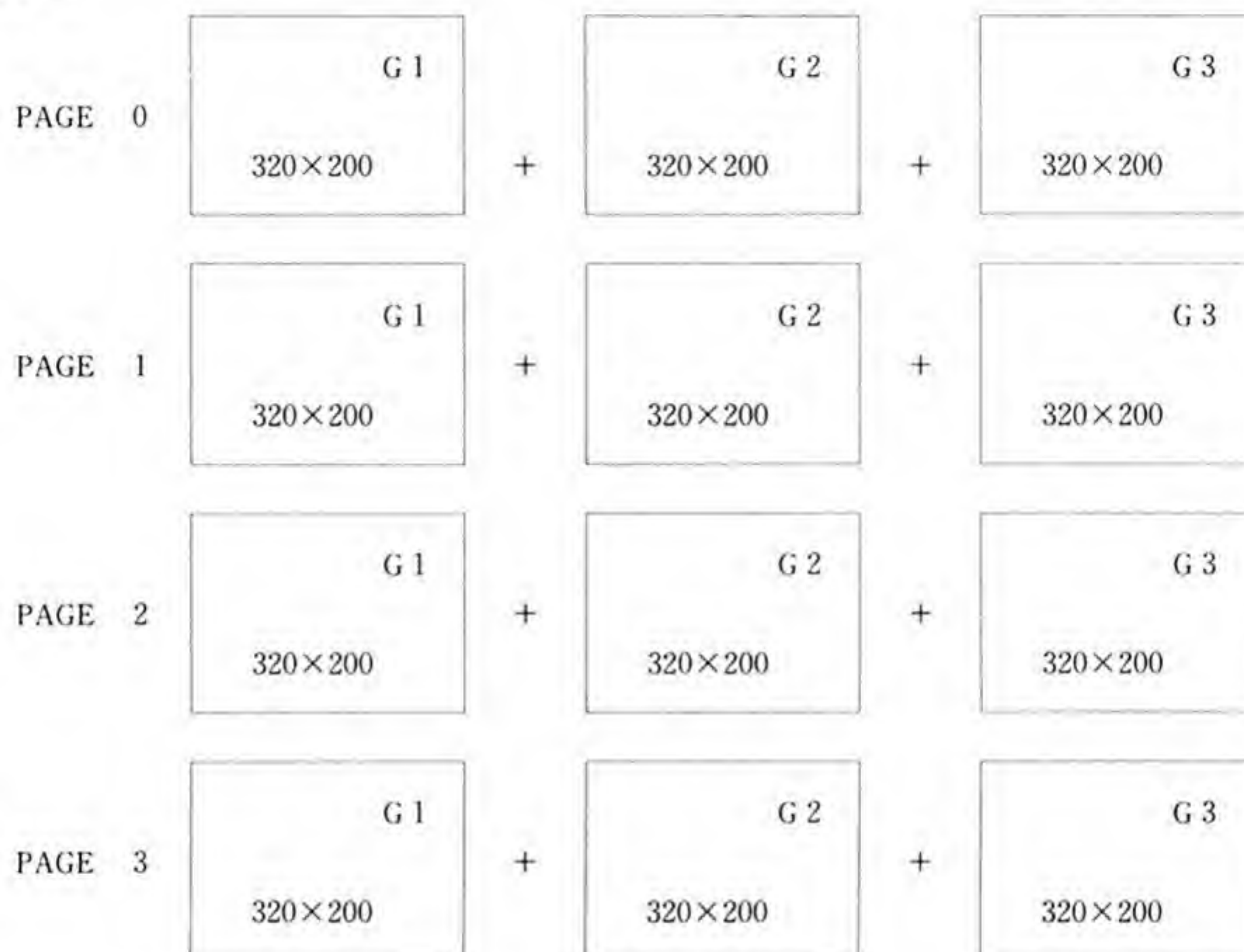
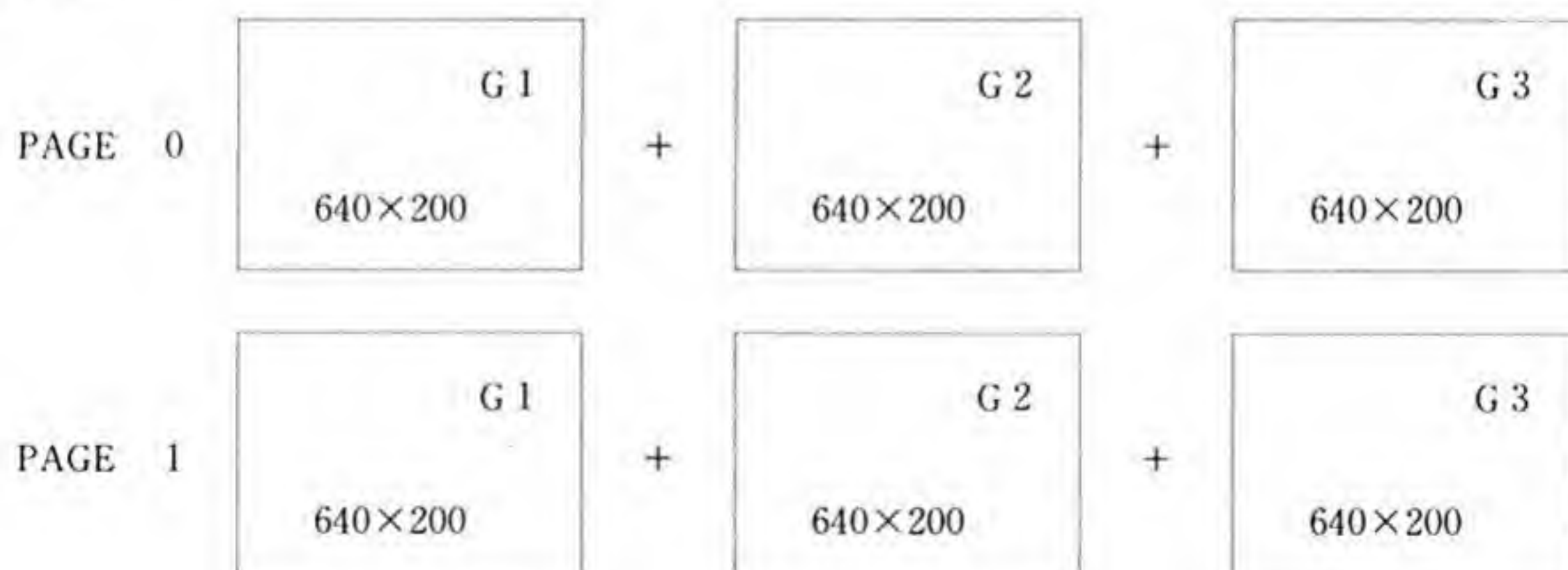
ステートメント	解像度	使用ページ数
WIDTH40, 25, 0, d	320×200	カラー4 ／ 白黒12
WIDTH80, 25, 0, d	640×200	カラー2 ／ 白黒6
WIDTH40, 12, 0, d	320×192	カラー4 ／ 白黒12
WIDTH80, 12, 0, d	640×192	カラー2 ／ 白黒6
WIDTH40, 25, 0, D	320×200	カラー4 ／ 白黒12
WIDTH80, 25, 0, D	640×200	カラー2 ／ 白黒6
WIDTH40, 12, 0, D	320×192	カラー4 ／ 白黒12
WIDTH80, 12, 0, D	640×192	カラー2 ／ 白黒6
WIDTH40, 25, 1, D	320×400	カラー2 ／ 白黒6
WIDTH80, 25, 1, D	640×400	カラー1 ／ 白黒3
WIDTH40, 12, 1, D	320×384	カラー2 ／ 白黒6
WIDTH80, 12, 1, D	640×384	カラー1 ／ 白黒3

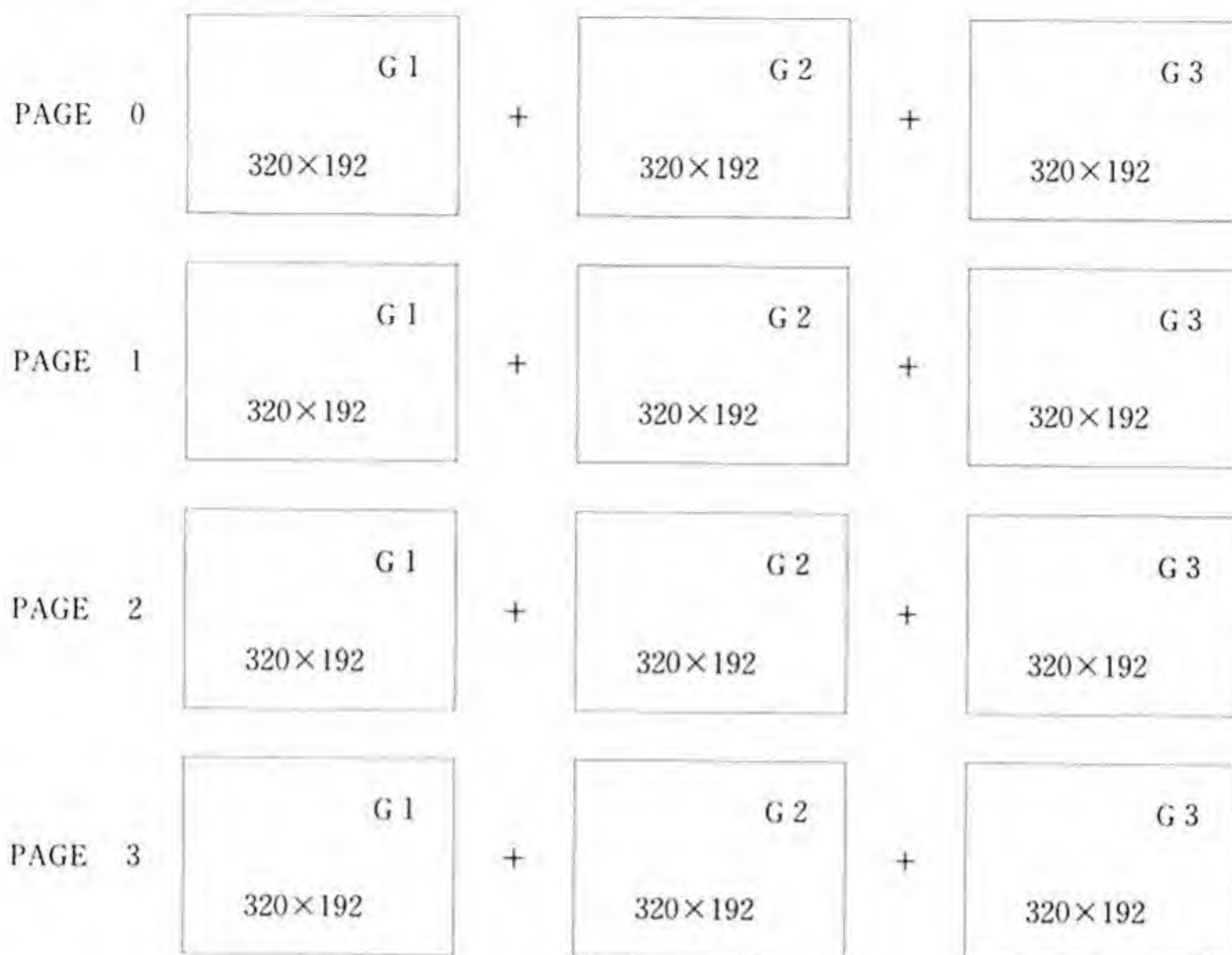
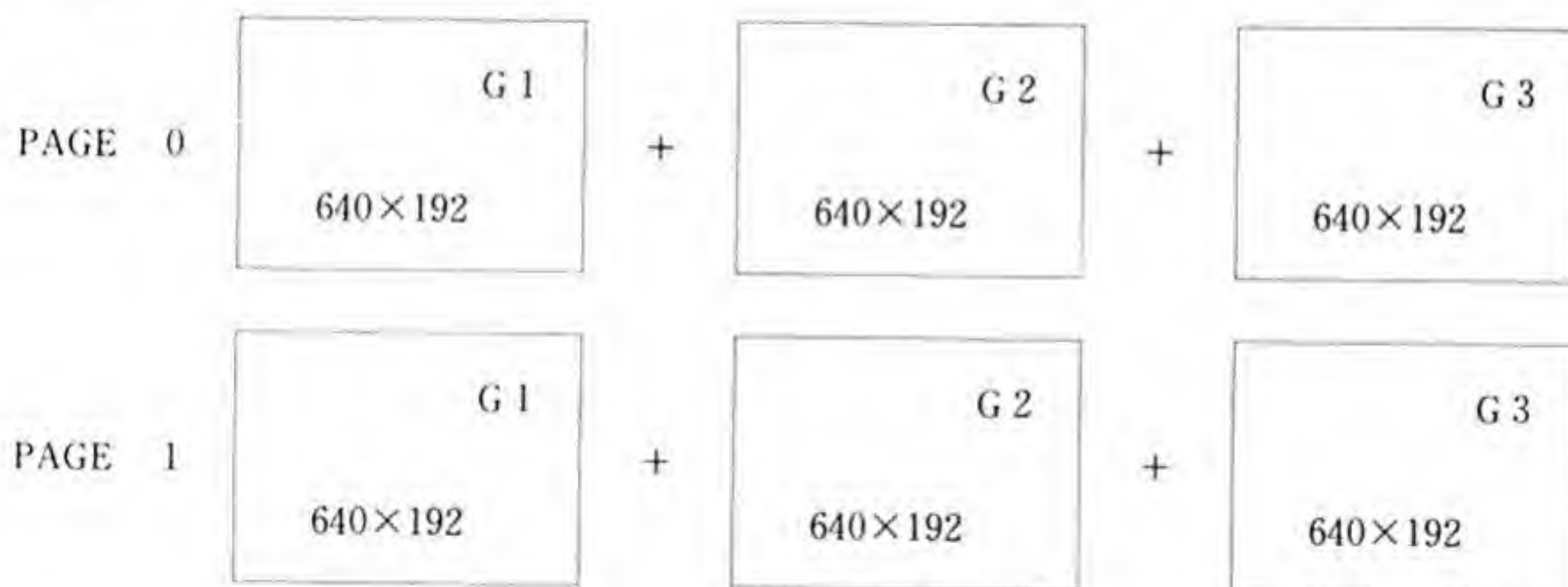
d：0または1（標準ディスプレイモードに設定。d＝0のとき、本体の標準／高解像度切換スイッチは標準ディスプレイモード（▬）にセットされていること）

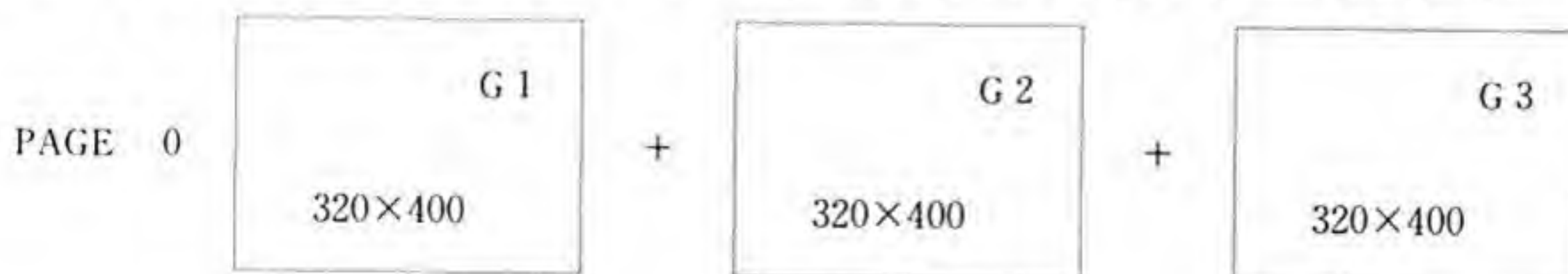
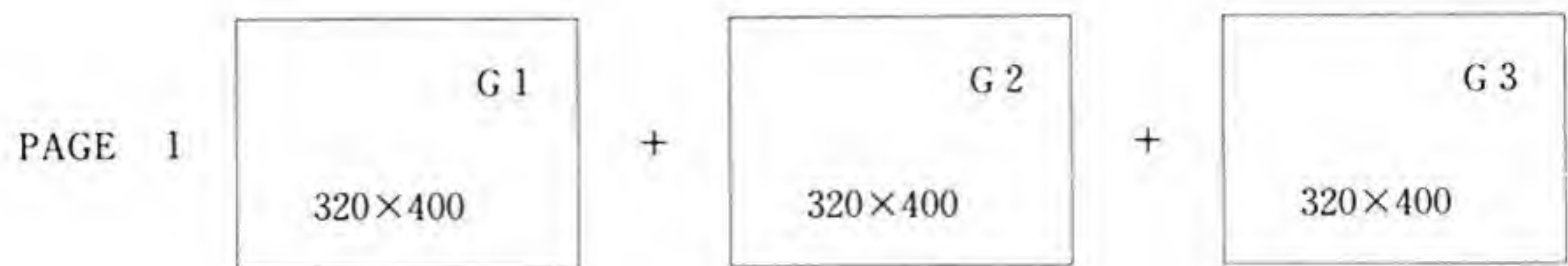
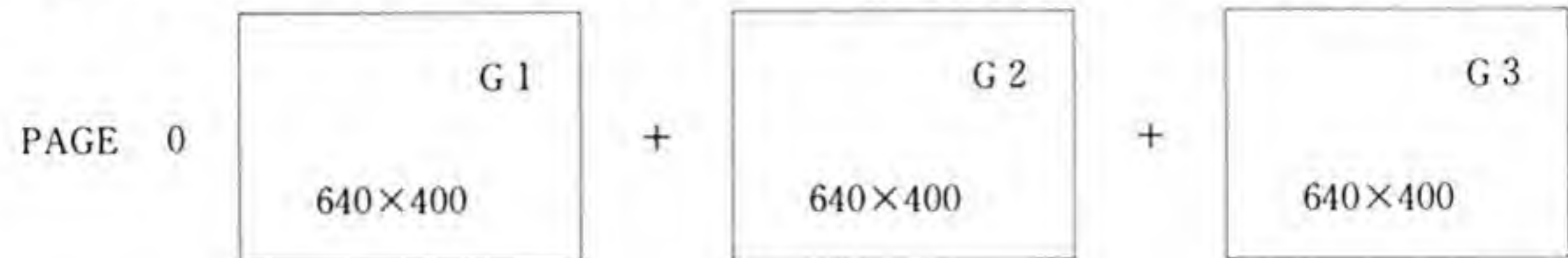
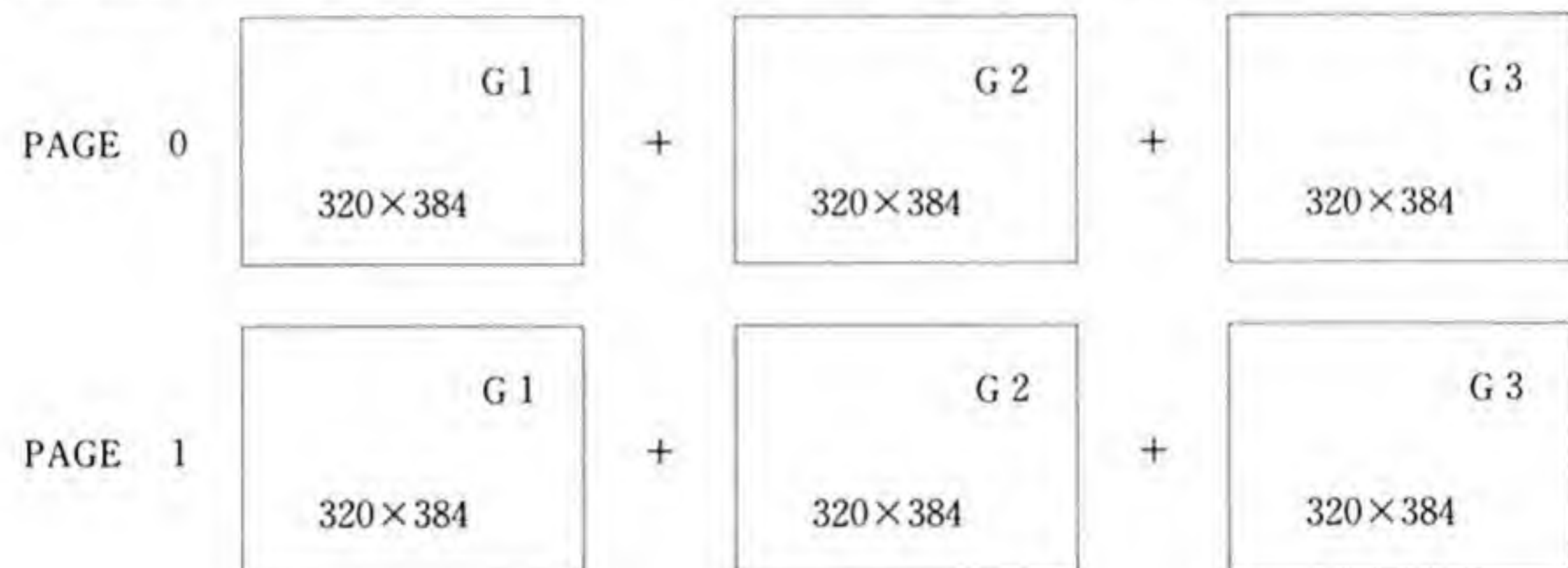
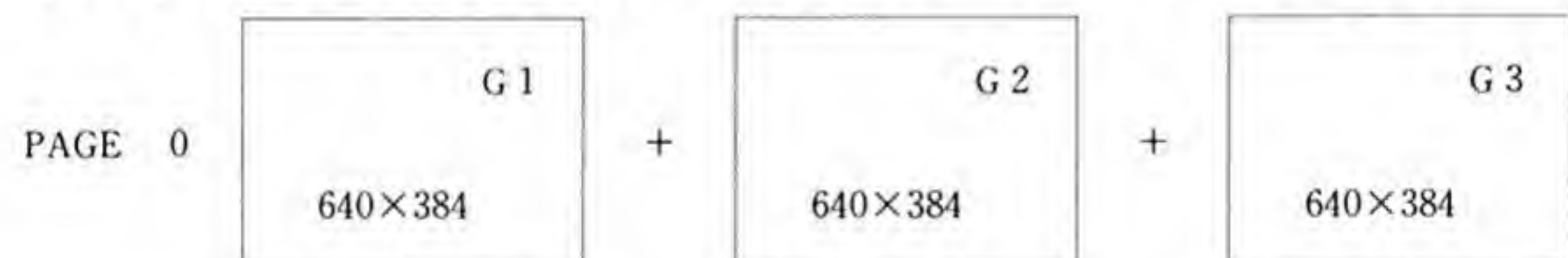
D：0または2（高解像度ディスプレイモードに設定。D＝0のとき、本体の標準／高度解像度切換えスイッチは高解像度ディスプレイモード（■）にセットされていること）

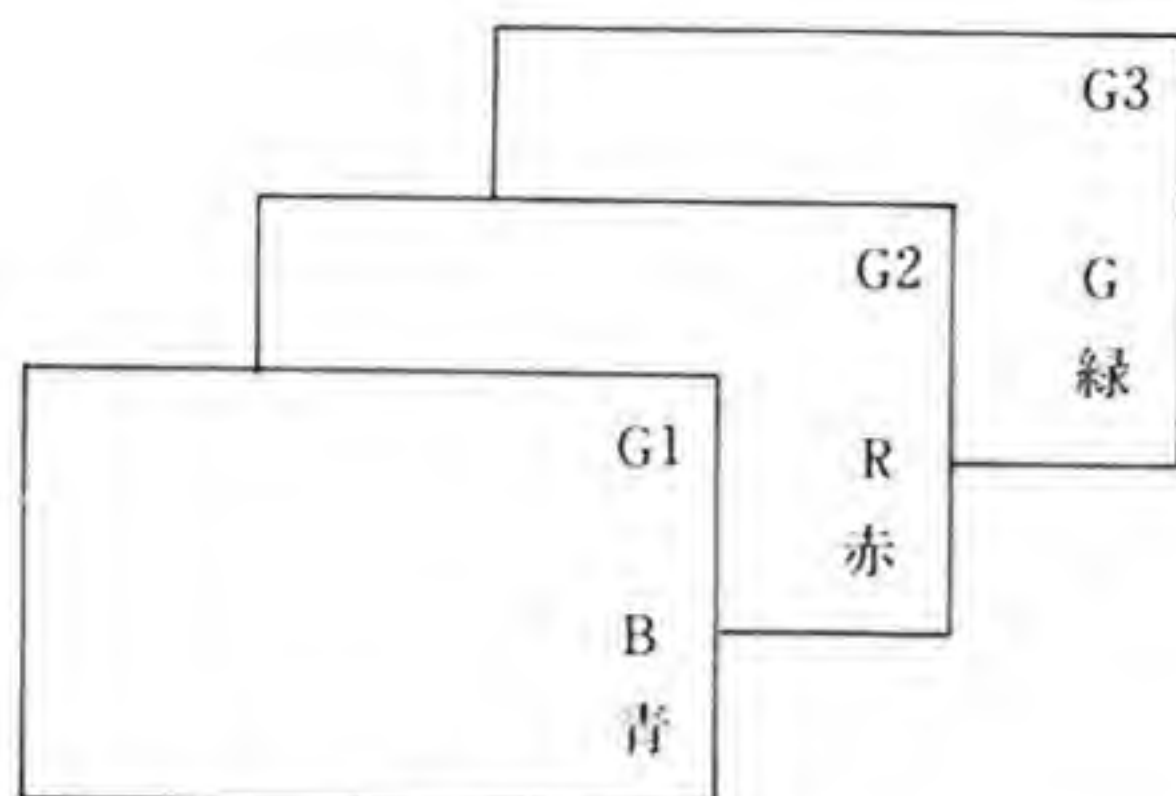
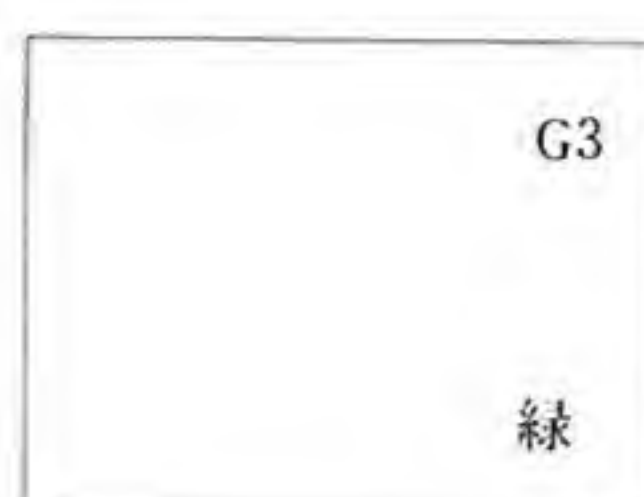
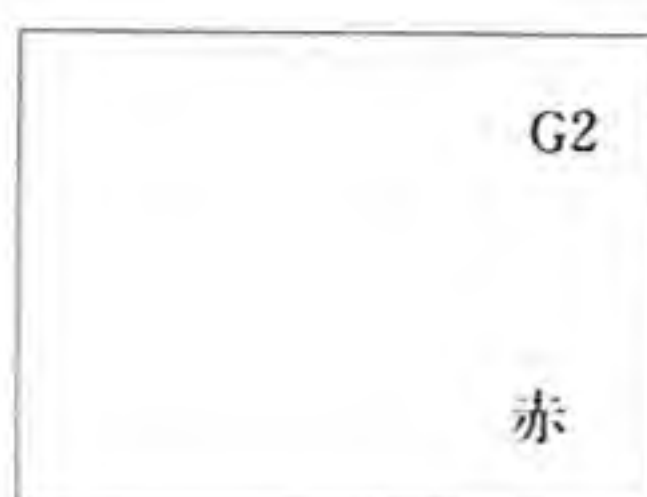
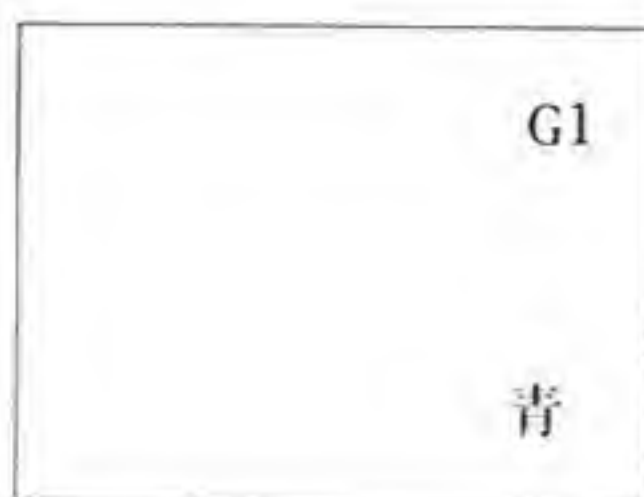
グラフィック画面の基本的な構成図を次に示します。

本書では、１枚目のグラフィック画面をＧ１、２枚目の画面をＧ２、３枚目の画面をＧ３と呼んで区別しています。

〈高解像度／標準ディスプレイモードで、ＷＩＤＴＨ４０，２５，０，Ｄを設定（Ｄ＝０または１または２）〉

〈高解像度／標準ディスプレイモードで、ＷＩＤＴＨ８０，２５，０，Ｄを設定（Ｄ＝０または１または２）〉

〈高解像度／標準ディスプレイモードで、ＷＩＤＴＨ４０，１２，０，Ｄを設定（Ｄ＝０または１または２）〉

PAGE 0	G 1 320×192	+	G 2 320×192	+	G 3 320×192
PAGE 1	G 1 320×192	+	G 2 320×192	+	G 3 320×192
PAGE 2	G 1 320×192	+	G 2 320×192	+	G 3 320×192
PAGE 3	G 1 320×192	+	G 2 320×192	+	G 3 320×192

〈高解像度／標準ディスプレイモードで、ＷＩＤＴＨ８０，１２，０，Ｄを設定（Ｄ＝０または１または２）〉

〈高解像度ディスプレイモードで、ＷＩＤＴＨ４０，２５，１，Ｄを設定（Ｄ＝０または２）〉

PAGE 0	G 1 320×400	+	G 2 320×400	+	G 3 320×400

〈高解像度ディスプレイモードで、ＷＩＤＴＨ８０，２５，１，Ｄを設定（Ｄ＝０または２)〉

〈高解像度ディスプレイモードで、ＷＩＤＴＨ４０，１２，１，Ｄを設定（Ｄ＝０または２)〉

〈高解像度ディスプレイモードで、ＷＩＤＴＨ８０，１２，１，Ｄを設定（Ｄ＝０または２)〉

※　画面内の積は、(横のドット数）×（たてのドット数）を表わす。

Ｇ１、Ｇ２、Ｇ３の各グラフィック画面は、次の図のようにＧ１が青、Ｇ２が赤、Ｇ３が緑というように設定されています。この３枚のグラフィック画面は合成されて画面上に表示されます。たとえば、青の画面にドットがあり、赤と緑の画面にドットがない場合はそのドットは青で表示されます。また、青と赤の画面にドットがあり、緑の画面にドットがない場合は、そのドットはマゼンタ（紫）で表示されます。
（カラーテレビの画面は､青､赤､緑の３原色を合成したものですが､ＢＡＳＩＣのカラーグラフィックも、同様に、青、赤、緑の３枚の画面を重ねて表示したものです。)

3枚のグラフィック画面の合成
(BRG合成)

3 カラーコード

カラーコードは0～7の整数で表わされ、その値によって画面の色が設定されます。

次の図はG1、G2、G3の画面とカラーコードの関係を示しています。

このようにカラーコードとは、青の画面をビット0、赤の画面をビット1、緑の画面をビット2とした3ビットの2進数を10進数表現したものとなっています。

色	G3　G2　G1	2進数ビット表現	カラーコード
黒（透明）	○　○　○	000	0
青	○　○　●	001	1
赤	○　●　○	010	2
マゼンタ	○　●　●	011	3
緑	●　○　○	100	4
シアン	●　○　●	101	5
黄	●　●　○	110	6
白	●　●　●	111	7

※　●はドットがある、○はドットがないことを示しています。

4 パレットコード

カラーコードは色そのものを指定するコードで、

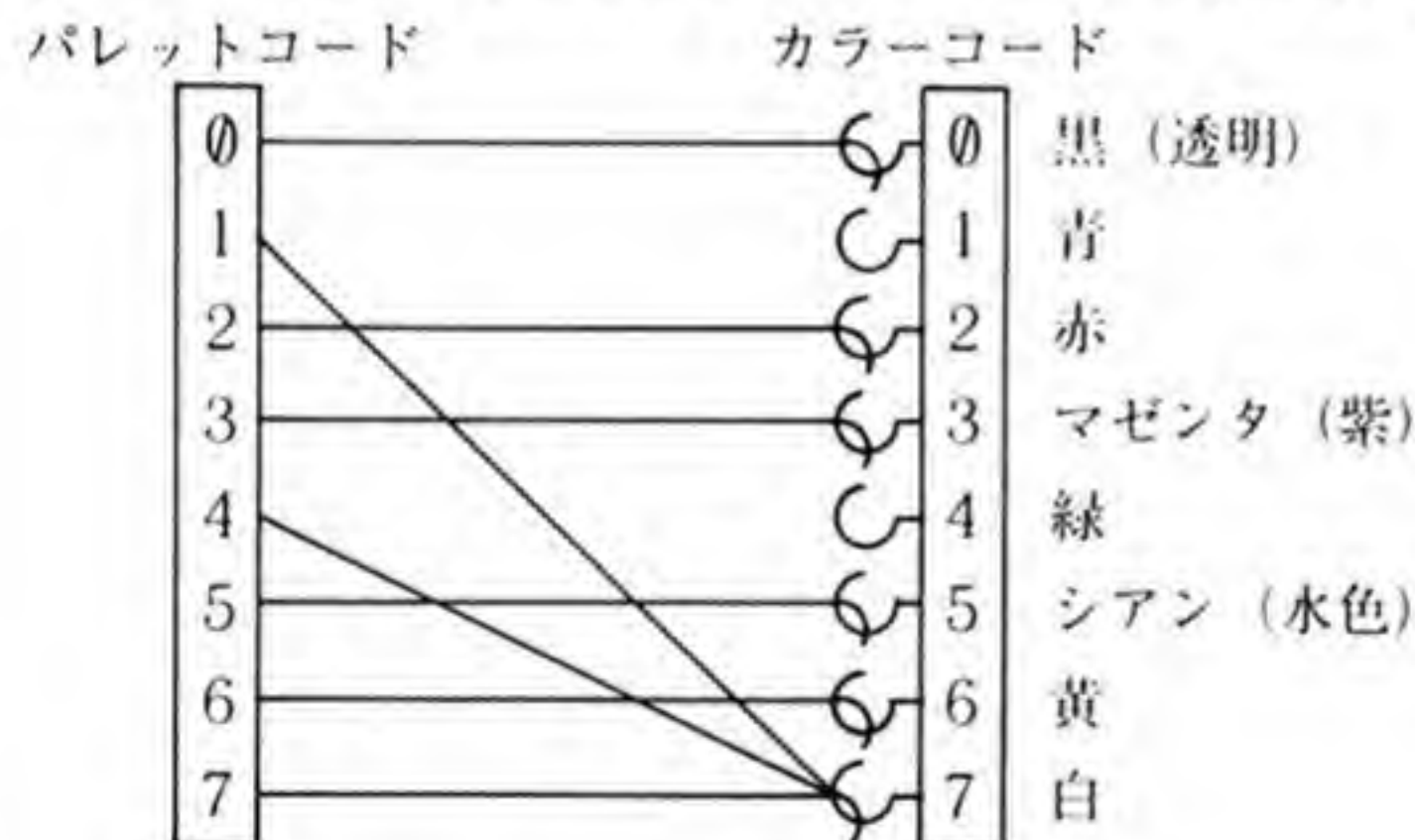
0＝黒（透明）、1＝青、2＝赤、3＝マゼンタ、4＝緑、5＝シアン、6＝黄、7＝白

というように、数字と色が絶対的に対応しています。

これに対してパレットコードも色を指定するコードで0～7の数字で表わされますが、カラーコードのように数字に対する色がきまっていませんので、自由に色を設定することができます。

パレットコードの色の設定はＰＡＬＥＴステートメントを使って行ないます。

次の図はパレットコードの設定の概念を示すものです。

ＢＡＳＩＣが起動した最初の状態は上のようにパレットコード＝カラーコードとなっていますが、次のようにフックをかけなおすと、パレットコードの1と4が白として使えるようになります。

5 中間色コード

グラフィックステートメントのうち

ＬＩＮＥのＢＦ（ボックスフィル）

ＰＡＩＮＴ

ＳＹＭＢＯＬ

の3つに対して、パレットコードを指定する代わりに「中間色コード」を指定することができます。

中間色コードは、

&Ｈｍｎ（またはｍ＊１６＋ｎ）　ｍ＝0～7のパレットコード

ｎ＝0～Ｆのコード

の１６進数2けたで表わされ、ｍとｎの2色を混合した中間色を出すことができます。

ただし、ｍとｎが同じパレットコードの場合は、パレットコードｍかｎを単独で指定したときの色になります。また、黒と他の色との混合色

&Ｈ00、&Ｈ01、………、&Ｈ07

はカラーコードの

　　0、1、……、7

そのもので中間色を出すことができないので、黒と他の色との混合色は、

　　&H08、&H09、……、&H0F

で代用しています。

〈&Hmnと&Hnmの相違点〉

&H12と&H21はパレットコード1（初期状態が青）とパレットコード2（初期状態が赤）を混合した同じ中間色（初期状態で紫）を表わしますが、グラフィック画面上のドット構成は次のように全く反転しています。

〈中間色一覧表〉

中間色コードと表示される色との関係を次の表にまとめます。実際の色を確認する場合は表の下のプログラムを実行してください。

中間色コード　&Hmnまたはm＊16＋n

m＼n	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	黒 (0)	青 (1)	赤 (2)	マゼンタ (3)	緑 (4)	シアン (5)	黄 (6)	白 (7)	黒 (8)	ダークブルー (9)	茶 (10)	紫 (11)	深緑 (12)	青緑 (13)	黄土色 (14)	灰色 (15)
1	ダークブルー (16)	青 (17)	紫 (18)	青紫 (19)	青緑 (20)	空色 (21)	灰色 (22)	薄紫 (23)								
2	茶 (32)	紫 (33)	赤 (34)	ピンク (35)	黄土色 (36)	灰色 (37)	褐色 (38)	はだ色 (39)								
3	紫 (48)	青紫 (49)	ピンク (50)	マゼンタ (51)	灰色 (52)	薄紫 (53)	はだ色 (54)	薄ふじ色 (55)								
4	深緑 (64)	青緑 (65)	黄土色 (66)	灰色 (67)	緑 (68)	緑青 (69)	黄緑 (70)	ホワイトグリーン (71)								
5	青緑 (80)	空色 (81)	灰色 (82)	薄紫 (83)	緑青 (84)	シアン (85)	ホワイトグリーン (86)	水色 (87)								
6	黄土色 (96)	灰色 (97)	褐色 (98)	はだ色 (99)	黄緑 (100)	ホワイトグリーン (101)	黄 (102)	クリーム (103)								
7	灰色 (112)	薄紫 (113)	はだ色 (114)	藤ふじ色 (115)	ホワイトグリーン (116)	水色 (117)	クリーム (118)	白 (119)								

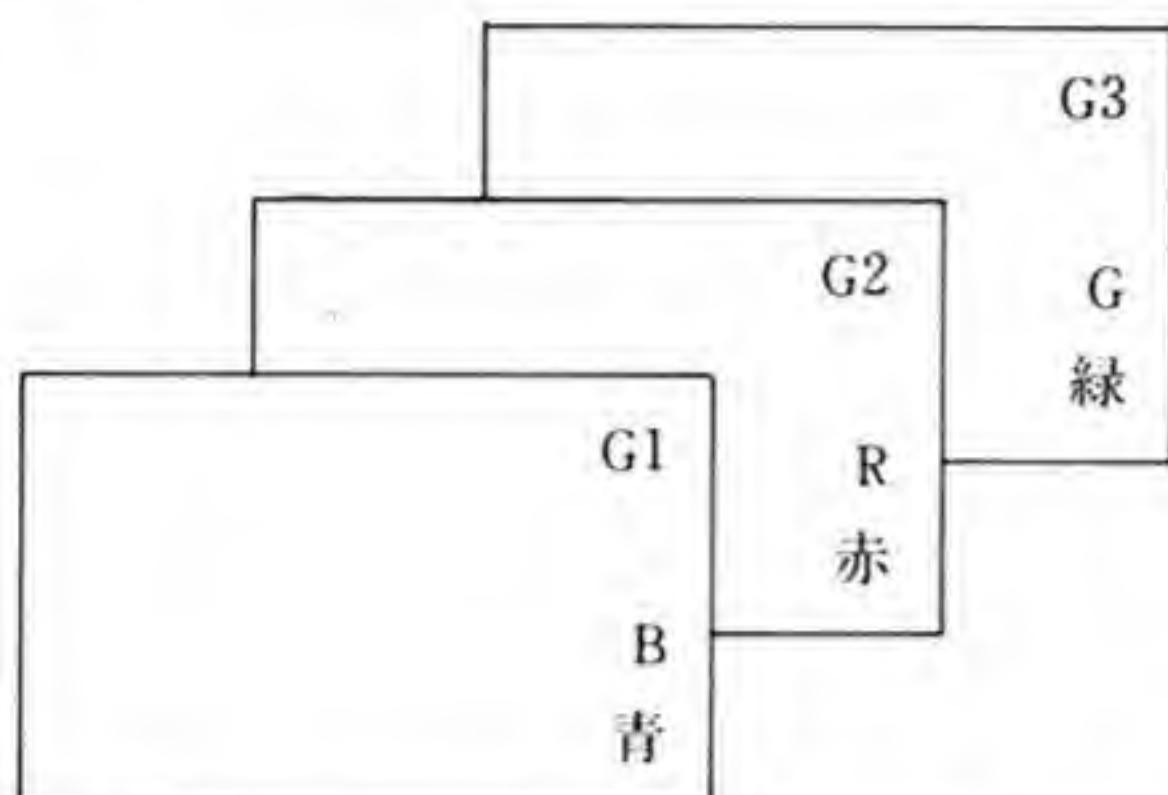
・色は、初期状態のパレットコードによって出されるもの。
・（　）内は10進数表現。

```
100 '中間色コード
110 INIT:WIDTH 80,12,0:CLS4
120 CSIZE2:PRINT#0,"  中間色コード(&Hmn or m*16+n)":CSIZE0:PRINT
130 PRINT"   m":CHR$(&H815F);"n   0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F"
;
140 FOR I=0 TO 7
150 LOCATE 3,I+3:PRINT I;
160 NEXT
170 FOR J=0 TO 7
180 FOR I=0 TO 15
190 LINE(67+I*32,48+J*16)-(96+I*32,62+J*16),PSET,J*16+I,BF
200 NEXT
210 NEXT
220 LOCATE 0,0:END
```

＊標準ディスプレイモードではグラフィック画面のドットが粗いので、中間色として見にくい部分があります。

6 カラーモード

カラーモードとは、「ＳＣＲＥＥＮ文の第３パラメータ（グラフィックモードの設定）が0となっていて、全グラフィック画面がアクセスできるモード」のことをいいます。

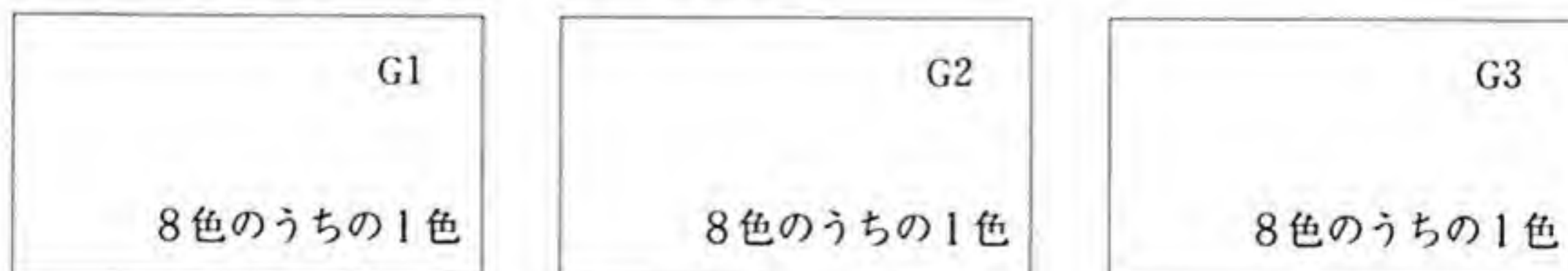
このモードのとき、グラフィック画面の各ドットに対して８色のうち任意の１色を指定することができます。

３枚のグラフィック画面を合成した画面で、８色のカラーを指定できる画面をカラー画面といいます。

画面の解像度の設定により、何ページかの独立したカラー画面を設定することができます。

解　像　度	カラー画面のページ数
320×200、320×192	4（PAGE0 ～ PAGE3）
640×200、640×192	2（PAGE0，PAGE1）
320×400、320×384	
640×400、640×384	1

カラー画面が複数ページであるときは、ＳＣＲＥＥＮステートメントの第１パラメータと第２パラメータで表示するページと書き込むページを指定することができます。

7 マルチページモード

マルチページモードとは、「ＳＣＲＥＥＮ文の第３パラメータ（グラフィックモード）の指定が１、２、３のいずれかになっていて、Ｇ１、Ｇ２、Ｇ３の３枚の画面を各々単色画面としてアクセスできるモード」のことをいいます。このモードのとき、ＣＡＮＶＡＳステートメントによって、各画面の色を任意の１色に設定することができます。

画面の解像度により、使用できる画面のページ数を設定することができます。

ステートメント	解像度	マルチ画面のページ数
WIDTH40, 25, 0, d	320×200	3×4
WIDTH40, 12, 0, d	320×192	
WIDTH80, 25, 0, d	640×200	3×2
WIDTH80, 12, 0, d	640×192	
WIDTH40, 25, 0, D	320×400	
WIDTH40, 12, 0, D	320×384	
WIDTH80, 25, 0, D	640×400	3×1
WIDTH80, 12, 0, D	640×384	

※d：0、1、2（標準ディスプレイモードまたは高解像度ディスプレイモードに設定。）
　D：0、2（高解像度ディスプレイモードに設定。D＝0のとき本体の標準／高解像度切換スイッチは、HIGH側(■)にセットされていること。）

8 ディスプレイ画面上の座標系

テキスト画面、グラフィック画面とも画面の左上が原点（0，0）であり、横方向がx軸、たて方向がy軸となっています。

8.1 テキスト座標系

テキスト座標系は、テキスト画面において設定されている座標系で、表示文字数によって次の図のようになっています。

〈WIDTH40,25,G,D（G＝0または1、D＝0または1または2）のとき〉

〈WIDTH80,25,G,D（G＝Øまたは1、D＝Øまたは1または2）のとき〉
（Ø，Ø）
（79，Ø）
（Ø，24）
（79，24）
〈WIDTH40,12,G,D（G＝Øまたは1、D＝Øまたは1または2）のとき〉
（Ø，Ø）
（39，Ø）
（Ø，11）
（39，11）
〈WIDTH80,12,G,D（G＝Øまたは1、D＝Øまたは1または2）のとき〉
（Ø，Ø）
（79，Ø）
（Ø，11）
（79，11）
〈WIDTH40,20,G,D（G＝Øまたは1、D＝Øまたは1または2））のとき〉
（Ø，Ø）
（39，Ø）
（Ø，19）
（39，19）
〈WIDTH80,20,G,D（G＝Øまたは1、D＝Øまたは1または2）のとき〉
（Ø，Ø）
（79，Ø）
（Ø，19）
（79，19）

〈WIDTH40,10,G,D（G＝0または1、D＝0または1）のとき〉

※D＝0のとき本体の標準／高解像度切換スイッチは標準ディスプレイモード(▬)にセットされていること。

〈WIDTH80,10,G,D（G＝0または1、D＝0または1）のとき〉

※D＝0のとき本体の標準／高解像度切換スイッチは標準ディスプレイモード(▬)にセットされていること。

CONSOLEおよびLOCATEステートメントは、前ページに示した座標の範囲で設定することができます。

8.2 グラフィック座標系

グラフィック座標系は、グラフィック画面において設定されている座標系です。
グラフィックの解像度によって、次のような座標の範囲が設定されます。

〈WIDTH40,25,0,D（D＝0または1または2）のとき〉

〈WIDTH40,25,1,（D＝0または2）のとき〉

※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード(■)にセットされていること。

<WIDTH80,25,0,D（D=0または1または2）のとき>

(0, 0)　(639, 0)

(0,199)　(639,199)

<WIDTH80,25,1,D（D=0または2）のとき>

(0, 0)　(639, 0)

(0,399)　(639,399)

※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード(■)にセットされていること。

<WIDTH40,12,0,D（D=0または1または2）のとき>

(0, 0)　(319, 0)

(0, 191)　(319,191)

<WIDTH40,12,1,D（D=0または2）のとき>

(0, 0)　(319, 0)

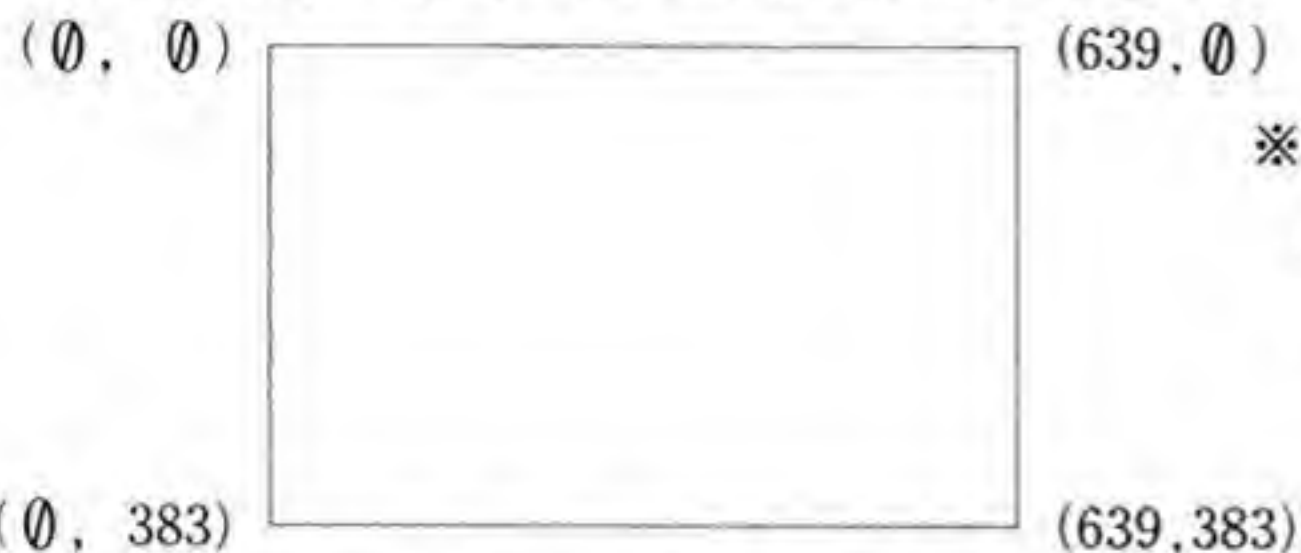
(0, 383)　(319,383)

※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード(■)にセットされていること。

<WIDTH80,12,0,D（D=0または1または2）のとき>

(0, 0)　(639, 0)

(0,191)　(639,191)

<WIDTH80, 12, 1, D (D=0または2) のとき>

※D=0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード(■) にセットされていること。

※WIDTH40,20,G,D (G=0または1、D=0または1または2)、
WIDTH80,20,G,D (G=0または1、D=0または1または2)、
WIDTH40,10,G,D (G=0または1、D=0または1)、
WIDTH80,10,G,D (G=0または1、D=0または1)
のときはグラフィック画面が使用できません。したがって、グラフィック座標系は設定されません。

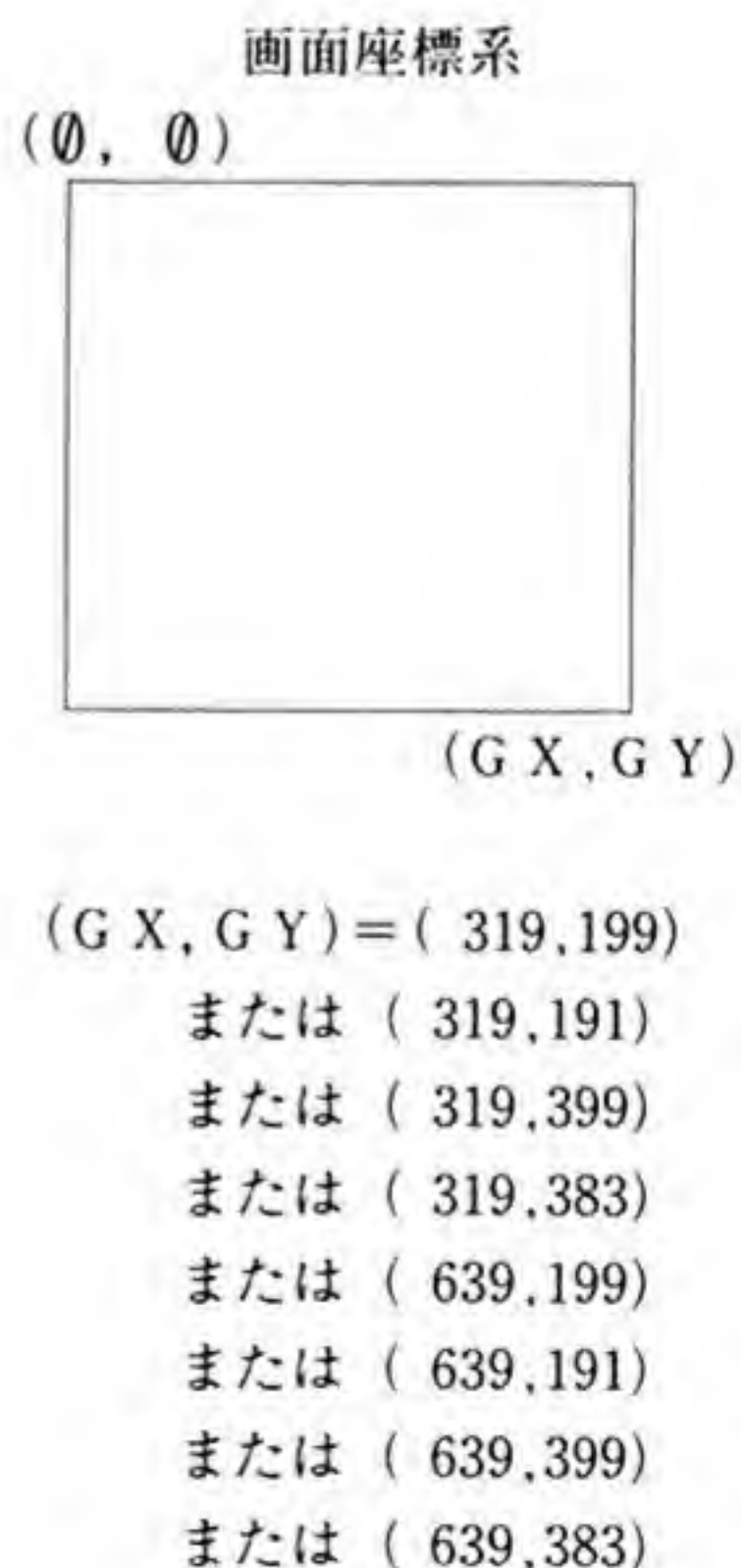
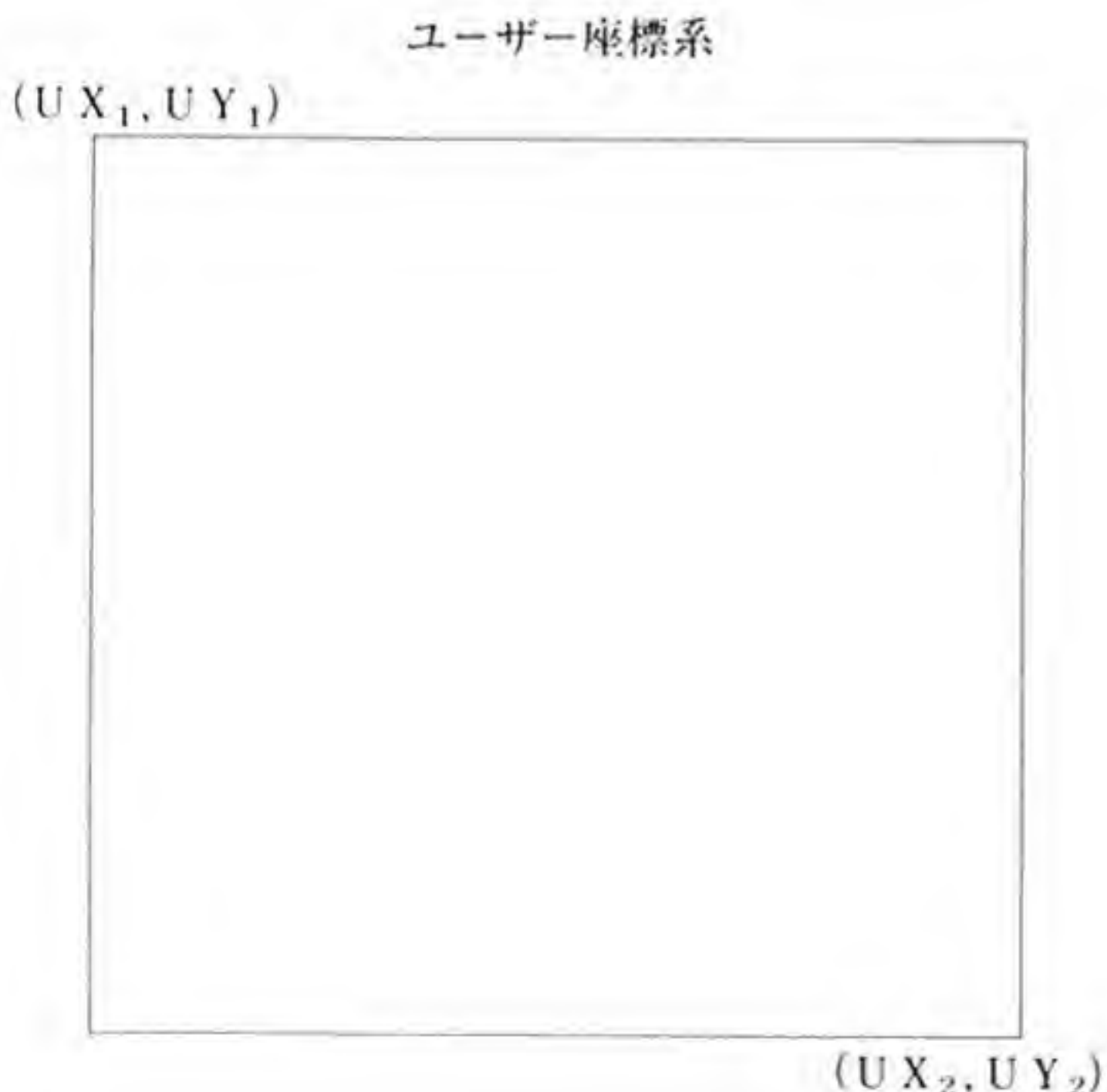
8.3 ユーザー座標系と画面座標系

ディスプレイ画面は、テキスト画面とグラフィック画面が重なって表示され、テキスト画面には**テキスト座標系**、グラフィック画面には**グラフィック座標系**が設定されています。このうちグラフィック座標系は**画面座標系**と**ユーザー座標系**という2つの座標系をもっています。

画面座標系とは、グラフィック座標系で説明した座標に一致するもので、画面上の1ドットが座標単位の1に対応します。

ユーザー座標系とは、BASICのWINDOWステートメント（『BASICリファレンスマニュアル』の2.8.2 WINDOW参照）によってユーザーが指定した座標系で、たて方向、横方向とも−1.7014118E+38〜1.7014118E+38（指数部−38〜+38）の単精度の範囲で指定できます。PSET, PRESET, LINE, POLY, CIRCLE, PAINTのグラフィックステートメント、およびPOINT関数はこのユーザー座標で使用されます。

下の図は、ユーザー座標と画面座標系です。

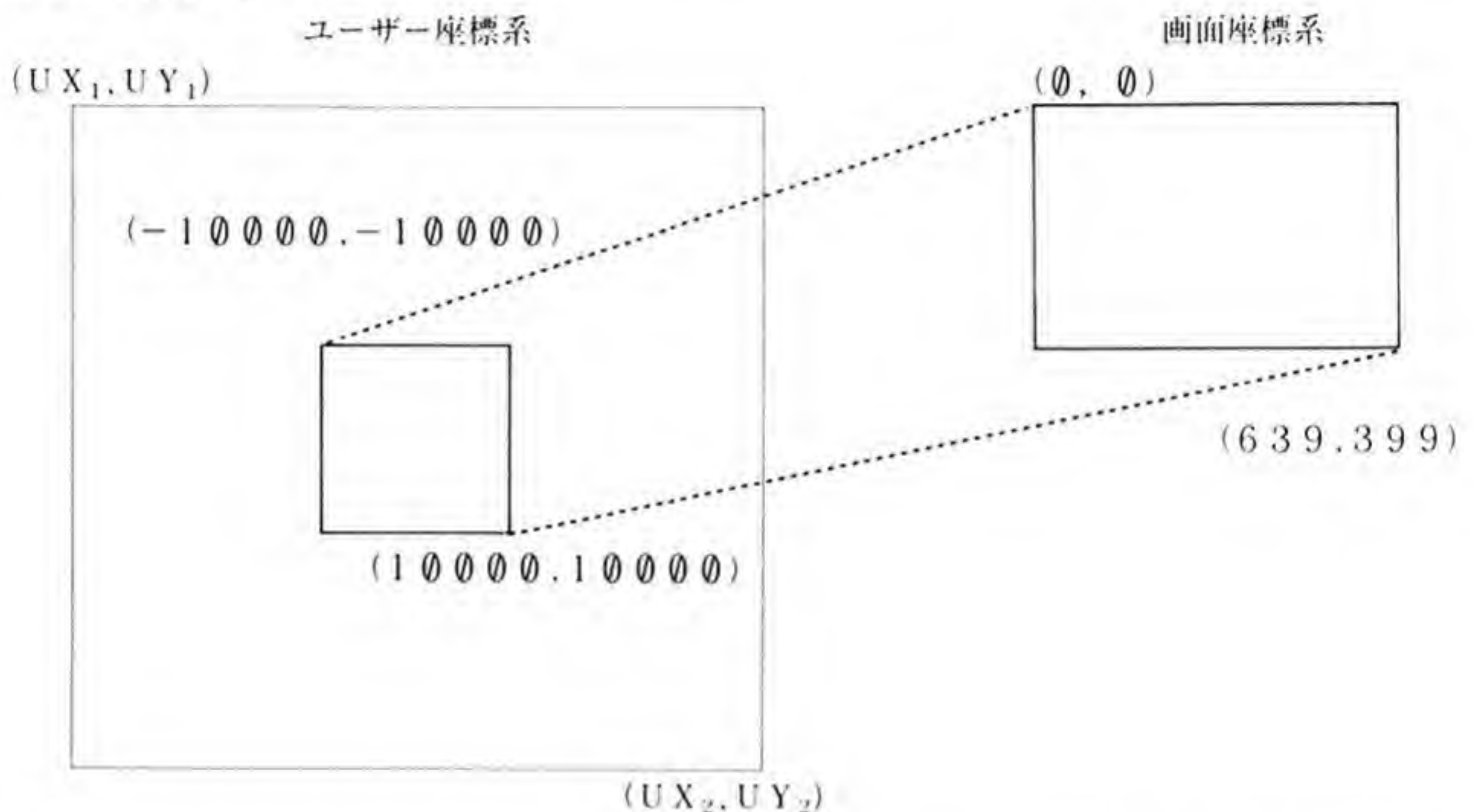
(GX, GY)=(319,199)
または (319,191)
または (319,399)
または (319,383)
または (639,199)
または (639,191)
または (639,399)
または (639,383)

$(UX_1, UY_1) = (-1.7014118E+38, -1.7014118E+38)$
$(UX_2, UY_2) = (+1.7014118E+38, +1.7014118E+38)$

画面座標系は実際に目で見ることができますが、ユーザー座標系の広大な座標平面は目で見ることができません。

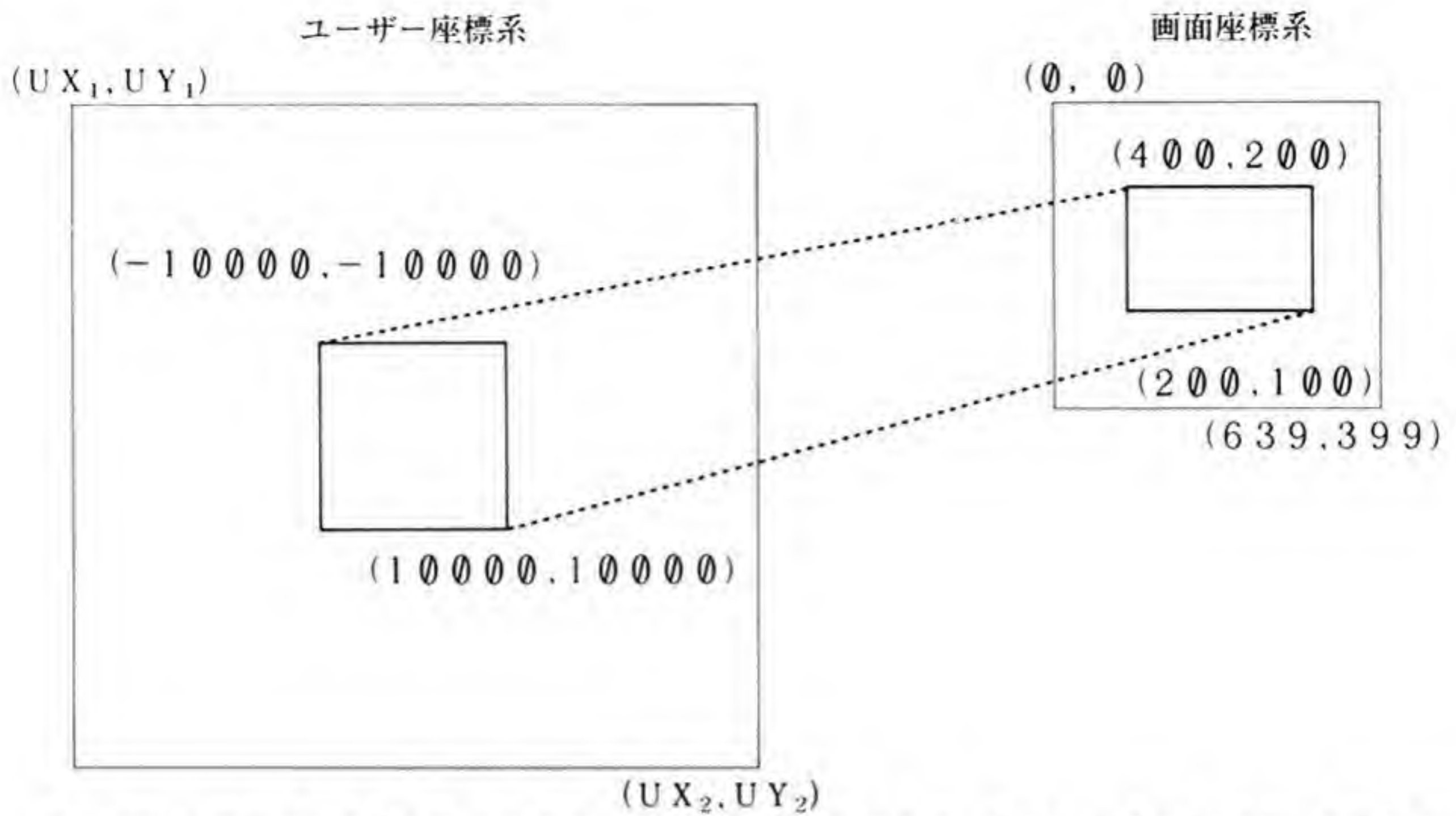
したがって、ＢＡＳＩＣのＷＩＮＤＯＷステートメントを使って、ユーザー座標系の座標平面中の任意の領域を画面座標系の平面上に割り当てなければなりません。

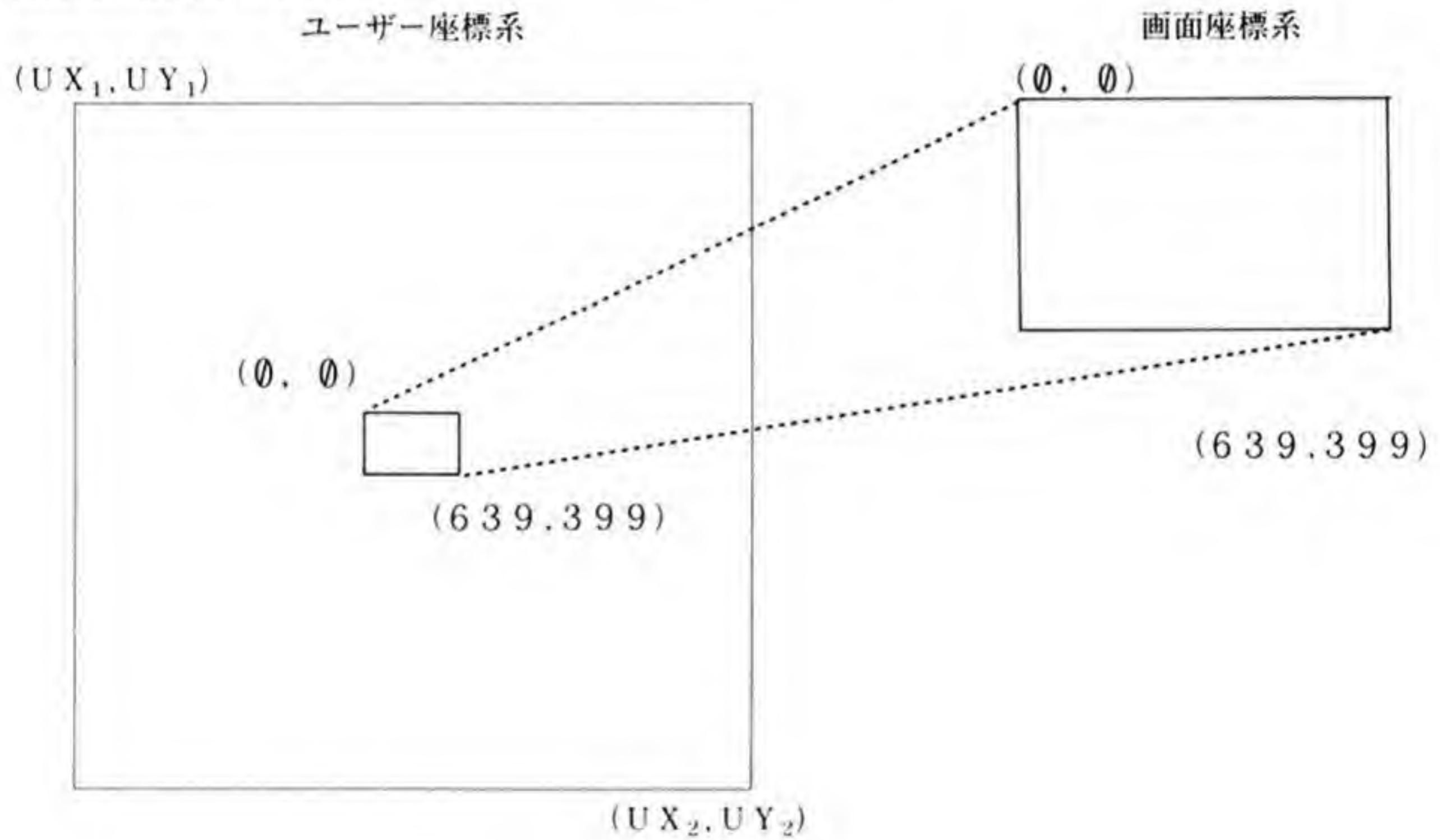
たとえば、ＷＩＤＴＨ８０，２５，１，２が設定されているとき、

ＷＩＮＤＯＷ（０，０）－（６３９，３９９），（－１００００，－１００００）－（１００００，１００００）⏎

と実行すると、次のようにユーザー座標系の（－１００００，－１００００）から（１００００，１００００）を対角線とする長方形の領域を画面座標系全域（グラフィック画面全体）に表示することができます。

また、

ＷＩＮＤＯＷ（２００，１００）－（４００，２００），（－１００００，－１００００）－（１００００，１００００）⏎

を実行すると、ユーザー座標系の（－１００００，－１００００）から（１００００，１００００）を対角線とする長方形の領域を画面座標系の（２００，１００）から（４００，２００）を対角線とする長方形の領域に表示することができます。

後ろのパラメータを省略して、ＷＩＮＤＯＷのみを実行すると、画面座標系の範囲と同じ領域が自動的に指定されます。

3章
テキスト

キーボードから入力された文字やプログラムのリストはディスプレイ画面に表示されます。このとき、もしディスプレイ画面に円や線などのグラフィックが描かれていれば、その上に重なって文字が表示されます。ここで画面消去の命令　ＣＬＳ⏎を入力すると文字のみが消去され、グラフィックはそのまま残って表示されています。グラフィックを描く画面と文字を表示する画面は独立しており、それぞれ**グラフィック画面**と**テキスト画面**といいます。

1 テキスト画面

テキスト画面にはアルファベット、数字、漢字、カタカナ、ひらがな、セミグラフィック文字などが書かれます。

ＷＩＤＴＨ命令は画面に表示する文字数を設定します。さらに、ＣＯＮＳＯＬＥ命令を使用すると、表示画面をテキストの命令が有効な領域と無効な領域に分けることができます。

CONSOLE　たて方向の表示開始行，たて方向の表示行数，横方向の表示開始行，横方向の表示桁数

たとえば、横方向の最初の桁から２０桁とたて方向の最初の行から１０行の領域をテキストの命令が有効な領域に設定するには、

```
CONSOLE 0,10,0,20 ⏎
```

と入力します。すると、ＬＩＳＴやＦＩＬＥＳ命令を実行した場合、この設定された領域（２０文字×１０行）内でリストやファイル名が表示され、それ以外の領域には表示されません。このようにして、テキスト表示領域内で、表示文字の変化可能領域と変化不可能領域とに分けることができます。この画面分割を解除するには、再度ＣＯＮＳＯＬＥ命令を実行して表示画面全体を指定するかまたは[CTRL]+[D]キーと押してください。

また、テキスト画面の任意の位置に文字を表示したりする場合にはＬＯＣＡＴＥ命令を使用します。

LOCATE　横方向の位置，たて方向の位置

たとえば、横方向１０桁目、たて方向１０行目の位置からＡＢＣという文字を表示させるには

```
LOCATE 10,10:PRINT"ABC"⏎
```

と入力します。

1.1 ファンクションキーの内容表示

テキスト画面の最下位は、ファンクションキーの内容表示に使用することができます。

ただし、表示画面の最下行は、CONSOLEステートメントによりテキストへの命令が無効領域に設定されている必要があります。

ファンクションキーの表示は　KEY　LIST　という命令を使って行ないます。

```
KEY LIST 0 ⏎
```

と入力するとファンクションキーの内容は表示されず、

```
KEY LIST 1 ⏎
```

と入力すると表示されます。

1.2 漢字とセミグラフィックパターン

本機は、漢字をテキスト画面に表示します。漢字を表示する場合画面モードを漢字表示のモードに設定しておく必要があります。この設定を行なうには、KMODEという命令を使います。

```
KMODE 0}
      1}
```

```
KMODE 1 ⏎
```

と入力すると漢字表示モードになります。

ただし、標準ディスプレイモードでたて方向の行数が25行または20行に設定されている場合には、漢字は表示できません。

このモードで「会社」という文字を表示するには

```
PRINT"会社"⏎
```

あるいは

```
PRINT CHR$(&J3271,&J3C52)⏎
```

と入力します。また、

```
PRINT CHR$(&H89EF,&H8ED0) ⏎
```

と入力しても

会社

と表示されます。

漢字コードは内部では2バイトで表現されており、最初の1バイトが&H80～&H9F　もしくは、&HE0～&HFFではじまるコードとなっています。&H80～&H9F，&HE0～&HFFのコードはセミグラフィックパターンのコードと重複しています。したがって漢字とセミグラフィックパターンとは同時に表示できません。そこで漢字の代りにセミグラフィックパターンを表示するには

```
KMODE 0 ⏎
```

と入力します。ここで

```
PRINT CHR$(&H89EF)⏎
```

と入力してみてください。今度は「会」という文字ではなく▮□というセミグラフィックパターンが表示されます。

2 テキストの属性

テキストは最大８０文字×２５行の２０００文字を表示することができますが、１文字単位で、色を指定することができます。また、反転や点滅も同様に１文字単位で行なえます。
この色や反転、点滅のことを**文字の属性**といいます。
そのほか文字の属性には倍文字、アンダーライン、ＲＯＭ／ＲＡＭキャラクタジェネレータがあります。
文字の属性は英数字、カナ、漢字すべてに指定できます。

2.1 色の指定

文字の色はＣＯＬＯＲという命令で行なえます。

```
COLOR 色の番号
```

０：黒、１：青、２：赤、３：マゼンタ、４：緑、５：シアン、６：黄、７：白
とすると、この命令が実行されてから以後に入力される文字は指定された色になります。
また、CTRLキーを押しながらテンキーの0〜7を押すことによっても文字の色指定ができます。

2.2 反転文字

文字は標準状態では指定された色で表示されていますが、反転モードにして入力すると、文字の色が補色になり白い四角で囲まれた状態になります。
補色は

青←→黄
赤←→シアン
緑←→マゼンタ
白←→黒

となります。

```
CREV {0または省略
      1}
```

反転モードにするには

CREV 1⏎

と入力します。
また

CREV 0⏎ もしくは CREV⏎

とすると標準モードにもどります。
また、CTRLキーを押しながらテンキーの－キーを押すことによっても反転モード、標準モードの切り換えができます。

2.3 点滅文字

点滅モードにすると、入力された文字は反転の状態と標準の状態を繰り返し表示します。

```
CFLASH { 0または省略 }
       {     1      }
```

点滅モードにするには

CFLASH 1⏎

とします。

標準モードにするには

CFLASH 0⏎ または **CFLASH**⏎

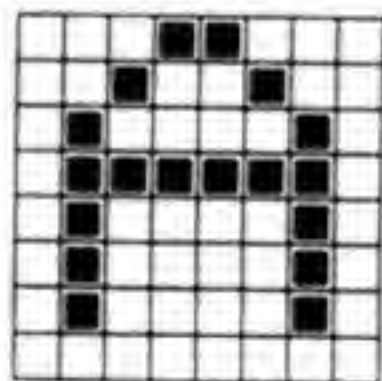
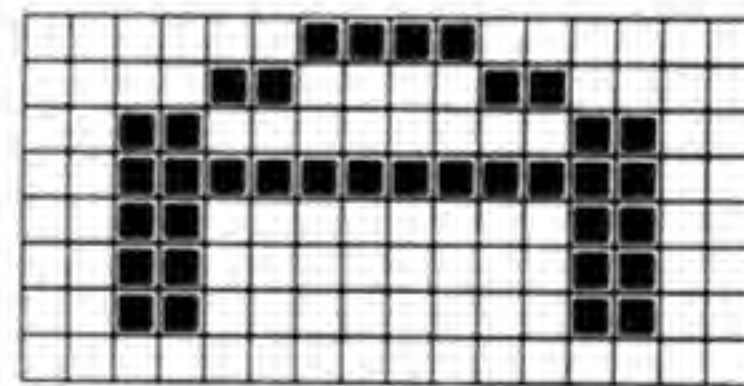
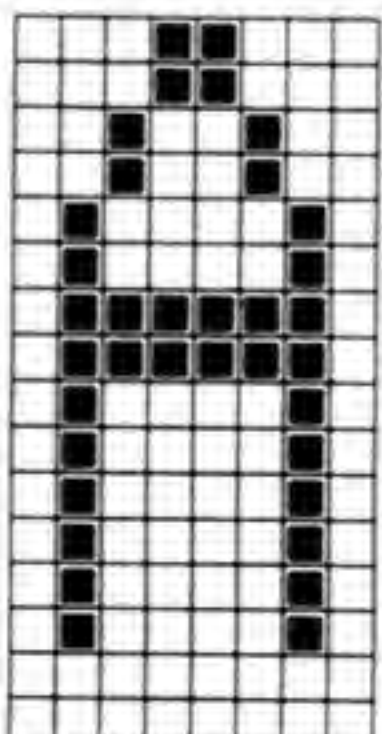
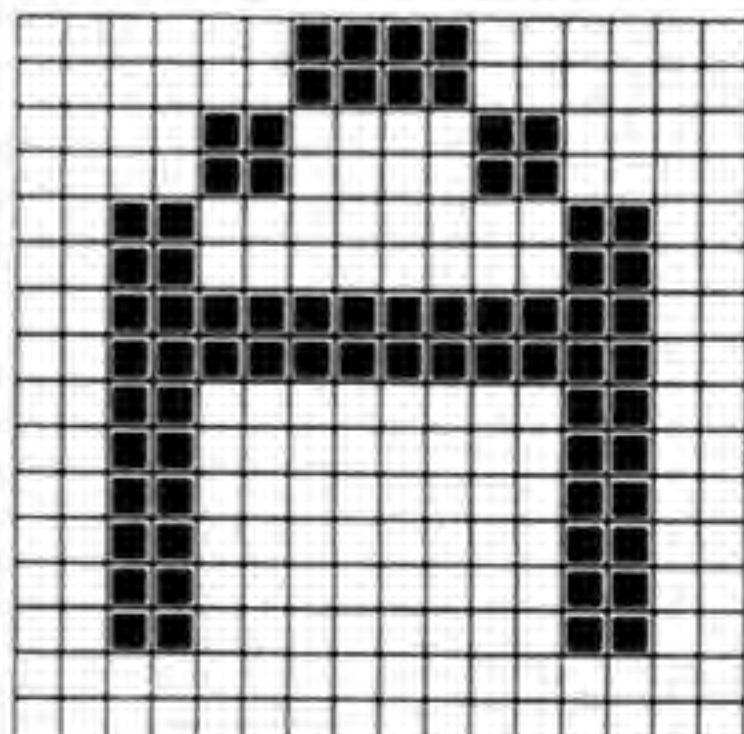
とします。また CTRL キーを押しながらテンキーの * キーを押すことによっても点滅モード、標準モードの切り換えができます。

2.4 倍文字

本機では4種類の文字を扱うことができます。これは、グラフィックで1ドットずつ文字を作るのではなくPRINT文で使えます。この4種類を区別して使いわける命令はCSIZEで、書式は次のとおりです。

```
CSIZE n
```

nには0、1、2、3のいずれかの数字を代入します。数字を代入した書式の内容は次のとおりです。

CSIZE0：標準の8×8ドットの文字です。
CSIZE1：縦方向に2倍の文字です。
CSIZE2：横方向に2倍の文字です。
CSIZE3：縦方向、横方向ともに2倍の文字です。

倍文字を書く場合、まずＣＳＩＺＥ文でサイズを指定します。次にＰＲＩＮＴ文を使いますが、ここで使うＰＲＩＮＴ文には＃０という記号を入れ書かせようとする文字列を続けて入れます。

```
PRINT#0, "〔文字列〕"
```

たとえば　10 CSIZE2
　　　　　20 PRINT#0,"ABC"とすると「ＡＢＣ」が横２倍の文字で書かれます。

文字を表示する座標の指定はＬＯＣＡＴＥ命令を使用します。文字の表示座標を指定する場合、次の点に注意してください。

① ＣＳＩＺＥ１（たて２倍）または、ＣＳＩＺＥ３（たて、横２倍）を指定した場合のたて方向の座標に関する注意点。

表示する文字より上にある他の文字との間隔は偶数行（０、２、４…行）あけなければなりません。もし上に何も文字がない場合には、偶数座標にしか表示できません。奇数座標に表示したい場合には、その座標より上方向に偶数行あけたスペースを挿入することにより可能となります。また、たて２倍あるいはたて、横２倍文字が表示されている２行内に、標準文字および横２倍文字を表示することはできません。

② ＣＳＩＺＥ２（横２倍）または、ＣＳＩＺＥ３（たて、横２倍）を指定した場合の横方向の座標に関する注意点。

横方向の座標を１、３、５、…の奇数座標に指定しても、指定した奇数座標に１を加えた２、４、６、…の偶数座標に文字は表示されます。よって、横２倍文字またはたて、横２倍文字を表示する場合の横方向の座標は０、２、４…の偶数座標を指定してください。

サンプルプログラム

ちょっとコマーシャル

文字の大きさを変えて"ＳＨＡＲＰ　ＰＥＲＳＯＮＡＬ　ＣＯＭＰＵＴＥＲ　Ｘ-１"と表示するものです。

```
50 REM CSIZE ヲ ツカウ
100 WIDTH 40 :INIT
110 CSIZE 3                              ←縦横2倍文字にします
120 CLS
200 LOCATE 5,8
210 COLOR 4
220 PRINT #0,"SHARP "                    ←縦横2倍文字で"SHARP"を表示
240 LOCATE 10,12
250 COLOR 2
260 CSIZE 1                              ←縦2倍文字にします
270 PRINT #0,"PERSONAL COMPUTER"         ←縦2倍文字で"PERSONAL COMPUTER"を表示
300 LOCATE 20,16
310 COLOR 6
320 CSIZE 2                              ←横2倍文字にします
330 PRINT #0,"X-1"                       ←横2倍文字で"X-1"を表示
400 END
```

2.5 ROMCGとRAMCG

テキスト画面に表示される文字はキャラクタジェネレータと呼ばれる部分から呼び出されたパターンが表示されますが、キャラクタジェネレータ（略してCG）はあらかじめ固定されたパターンが格納されているROMCGと、ユーザーが自由に定義できるRAMCGがあります。
ROMCGの中には、英数字、カナ、セミグラフィックなどのパターンが入っており、標準状態でキーを入力するとROMCGのパターンが表示されます。RAMCGの内容は電源を切ると消えてしまいます。したがって、RAMCGを使用する場合にはあらかじめRAMCGにパターンを定義しておかなければなりません。この定義に使用するステートメントが**DEFCHR$**です。また、RAMCGとROMCGとの表示を切り換えるステートメントが**CGEN**です。
そこで例をあげて説明します。

```
CGEN 0⏎
PRINT "A"⏎
```

とします。
すると画面にはAという文字が表示されます。
次に

```
DEFCHR$ (65) =STRING$ (8, &HFF) +STRING$ (8, &HF0)
                +STRING$ (8, &H81) ⏎
CGEN1⏎
PRINT"A"⏎
```

（CGEN1⏎以後はどのキーを押しても白ベタに表示されます。）
とすると画面には、縞模様のパターンが表示されます。
このパターンが　RAMCGにDEFCHR$で定義されたパターンです。
DEFCHR$（65）の65はASCIIコードを示しています。ASCIIコード65はROMCGの"A"にあたります。
RAMCGは8×8ドットのパターンで256個まで定義できます。
またRAMCGとROMCGとの表示を切り換えるには［CTRL］キーを押しながらテンキーの［/］を押すことによっても可能です。

2.6 アンダーライン

テキスト画面で、行と行の間にアンダーラインを表示することができます。
アンダーラインは画面に表示する行が20行もしくは10行のときにのみ設定することができます。
つまり、アンダーラインを使用する場合、あらかじめ

WIDTH, 20⏎　もしくは　**WIDTH, 10⏎**

としておきます。
これらのモードにすると、行と行の間にアンダーライン用のドットがあけて表示されます。
アンダーラインを表示する場合、**KSENステートメント**を使用します。

```
KSEN { 0 または 省略 }
     { 1           }
```

KSEN 1⏎

としますと、このステートメントを実行されたあとに入力された文字の下にアンダーラインが表示されます。

さらにアンダーラインの色は、KSEN命令の第2パラメータで指定できます。

KSEN 1, 色⏎

アンダーラインを表示しないモードにもどすには

KSEN 0⏎ または **KSEN⏎**

とします。

3 グラフィックと文字表示

LINE、SYMBOL、GET@、PUT@はグラフィックだけでなく文字表示にも使用できます。

たとえば

LINE (0, 0) − (39, 9), "A", B⏎

とすると(0, 0)と(39, 9)を対角線とするBOXを文字"A"でテキスト画面上に描きます。

また、

SYMBOL (0, 0), "ABCD", 1, 1, 7, 0, "#"⏎

とすると、"ABCD"という文字を"#"で描きます。

これらのステートメントの応用として、属性のみの変更に使用できます。

たとえば、80文字のモードで、画面の上から3行目の文字にすべてアンダーラインを付ける場合

KSEN 1, 7⏎
LINE (0, 2) − (79, 2), ""⏎

とします。(このとき、WIDTHステートメントでアンダーラインの表示可能な画面モードにしておきます) つまり、文字列のかわりに""(ヌルコード)を指定すると、その時点の属性で、LINEやSYMBOLを描くことができます。

次に、GET@はテキスト画面のデータを配列変数に取り込むことができます。

たとえば、画面(0, 0) − (5, 5)の範囲に表示されている文字をそのまま、(25, 15) − (30, 20)の位置に書く場合

DIM A (25)⏎
GET@(0, 0) − (5, 5), A⏎

とし、配列変数Aに文字を読み込みます。

そして、

PUT@(25, 15) − (30, 20), A⏎

とします。

4章

グラフィックス

本機の特長の１つに、豊富なグラフィック処理機能があげられます。９６Ｋバイトものグラフィック用メモリを標準装備していますので、複雑、多様な表現が可能です。本機のＢＡＳＩＣでは、多彩なグラフィック処理コマンドによって、これらの機能をサポートしています。本章では、多くのプログラム例をまじえながら、これらコマンド群の使い方を説明します。本章で解説するコマンド・ステートメントを次に示します。

4章

INIT、CLS、LINE、CIRCLE、POLY、PAINT、PALET、POSITION、PATTERN、SYMBOL、GET@、PUT@、PRW、COLOR

（注）本章は、**「２章、ディスプレイモード」**の内容を土台にして書かれています。本文中で何かわからない点がありましたら、２章の記述を参照してください。
なお、40×20、80×20、40×10、80×10行の各モードではグラフィックを表示できません。

1 画面の初期化

まず最初に、画面を初期化するステートメントについて説明します。コマンド・ステートメントの中には、画面状態の設定も変えてしまうものもありますので、そのままでは思い通りのグラフィックが書けない場合もあります。そのような時、次のステートメントを実行することによって、画面状態はもとの設定にもどります。

```
INIT
```

また、CTRL＋Dとダイレクトに入力することによっても、ＩＮＩＴステートメントと同様の結果が得られます。（CTRL＋Dは入門編１章「プログラミングを始めよう」参照）
上記ＩＮＩＴステートメントは、画面の設定状態を初期化する命令ですので、画面に描かれている内容をクリアすることはできません。画面をクリアする場合は、次のステートメントを実行します。

```
CLS 〔n〕    (n=0～4)
```

〔　〕で囲まれたパラメータは省略することができます。
ＣＬＳステートメントは、ｎの値によって処理が異なってきます。
ＣＬＳ０…………グラフィック画面Ｇ１、Ｇ２、Ｇ３(注)を同時にクリアします。
ＣＬＳ１…………グラフィック画面Ｇ１のみをクリアします。
ＣＬＳ２…………グラフィック画面Ｇ２のみをクリアします。

CLS3…………グラフィック画面G3のみをクリアします。
CLS4…………グラフィック画面G1、G2、G3およびテキスト画面を同時にクリアします。
CLS……………テキスト画面のみをクリアします。(SHIFT+CLR HOMEと同じ処理をします)。

注) グラフィック画面G1、G2、G3については、「2章　2. グラフィック画面」の項を参照してください。

上記INIT、およびCLSステートメントによって、画面状態の初期化とクリアが行なわれます。したがって、これからいろいろなコマンド、ステートメントを実行していく中で、画面状態を元に戻したい場合は、上記ステートメントを実行してください。

2 グラフィックステートメントの座標指定

グラフィック画面の座標系には、「2章　8. 3ユーザー座標系と画面座標系」で説明したように画面座標系とユーザー座標系の2種類あります。画面座標系は、画面上の1ドットが1座標に1対1に対応した絶対的な座標系ですので、不変ですが、ユーザー座標系はWINDOWステートメントの設定値によって変化します。これからこの章で述べていく各種グラフィックステートメントのグラフィック座標は、以上2つの座標系のどちらかに分類されますが、その分類の様子は次のようになります。

座標系	画面座標系	ユーザー座標系
グラフィック ステートメント	POSITION, SYMBOL, GET@, PUT@	PSET, PRESET, LINE, CIRCLE, POLY, PAINT

このようにグラフィックステートメントによって座標系が異なりますので、WINDOWステートメントを使用した場合は、まず、そのステートメントがどちらの座標系に所属しているかを調べてから座標指定を行なうようにしてください。ただし、BASIC起動時には、ユーザー座標系は、画面座標系とまったく同じものですので、WINDOWステートメントでユーザー座標系を変えない限り、両者の区別はありません。座標系の初期化、すなわち、ユーザー座標系を画面座標系と同一にするには、INITステートメントまたはCTRL+Dを実行します。

(注) WINDOWステートメントの使用方法については「2章　8.3ユーザー座標系と画面座標系」の項を参照してください。

3 グラフィックステートメントの色指定

カラーモードにおいてグラフィック画面は、1ドット単位に8色の色指定が可能です。これらの色指定には、カラーコードとパレットコードの2種類が用いられます。カラーコードは、色とコードが1対1に対応した絶対的なコードなので不変ですが、パレットコードは、色とコードの対応関係が、PALETステートメントによって変化します。したがって、これからこの章で述べていく各種グラフィックステートメントの色指定がどちらのコードで行なわれているか、あらかじめ知っておく必要があります。次に、その区別の様子を示します。

色指定	カラーコード	パレットコード
グラフィック　ステートメント	COLOR, CANVAS, KSEN	PSET, PRESET, LINE, CIRCLE, POLY, PAINT, SYMBOL, PRW

ただし，BASIC起動時には、パレットコードはカラーコードとまったく同じ色になっていますので、PALETステートメントでパレットコードの色指定を変えない限り、両者の区別はありません。パレットコードを初期化するためには、INITステートメントまたはPALETステートメントまたは CTRL + D を実行します。

また、グラフィックステートメントにおいて、色指定（パレットコード）を省略した場合は、自動的に、COLORステートメントの第1パラメータで指定した値（カラーコード）を、そのグラフィックステートメントのパレットコードとみなして実行します。

COLORステートメントの設定は、次の通りです。

```
COLOR 〔c1〕〔, c2〕
```

c_1：テキスト文字の表示色（カラーコード）

c_2：画面の背景色（カラーコード）

これより、グラフィックステートメント中の色指定を省略すると、パレットコードがカラーコードと等しい場合テキスト画面の文字の色と同じ色でグラフィックが描かれることになります。

たとえば、COLORステートメントの第1パラメータをカラーコード1に設定した場合、パレットコードが初期状態ならば、パレットコードの指定を省略したグラフィックステートメントは青色として実行されます。また、PALETステートメントによって、パレットコードの1がカラーコードの4に設定されていれば、パレットコードの指定を省略した、グラフィックステートメントは緑色として実行されます。なお、COLORステートメントは、BASIC起動時には、次のように設定されています。

```
COLOR 7, 0
```

また、COLORステートメントのかわりに、以下のようなダイレクト実行によっても同様な結果が得られます。

CTRL +テンキー 0　　(=COLOR 0)
CTRL +テンキー 1　　(=COLOR 1)
　　⋮　　　　　　　　⋮
CTRL +テンキー 7　　(=COLOR 7)

(注) PALETステートメントの設定方法については、後述「8. 色を瞬時に変える」の項を参照してください。

(注) 本章では、特に断わらない限り、カラーモードにおけるグラフィック表示について述べます。（カラーモードについては、「2章 6. カラーモード」の項を参照してください。）

4 線を描く

ＬＩＮＥステートメント

ＬＩＮＥステートメントは、ただ単に実線を引くだけではなく、パラメータの設定によって次の動作を行なうことができます。

①ラインスタイルを設定することによって、点線、破線、一点鎖線など自由な形のラインを引くことができます。

②任意のラインスタイルで、長方形を描くことができます。

③長方形を描いて、その中を任意のパレットコード、中間色コードまたはタイリングパターンで塗りつぶすことができます。

④座標ばかりを連続入力することによって、任意の折れ線や多角形を描くことができます。

4.1 線を引く

線を引く場合、ＬＩＮＥステートメントのパラメータは次のようになります。

```
LINE [(x1, y1)] - (x2, y2) [, mode, c, ls]
```

x_1、y_1：始点座標（ユーザー座標系）

x_2、y_2：終点座標（ユーザー座標系）

mode：ＰＳＥＴ、ＰＲＥＳＥＴ、ＸＯＲ

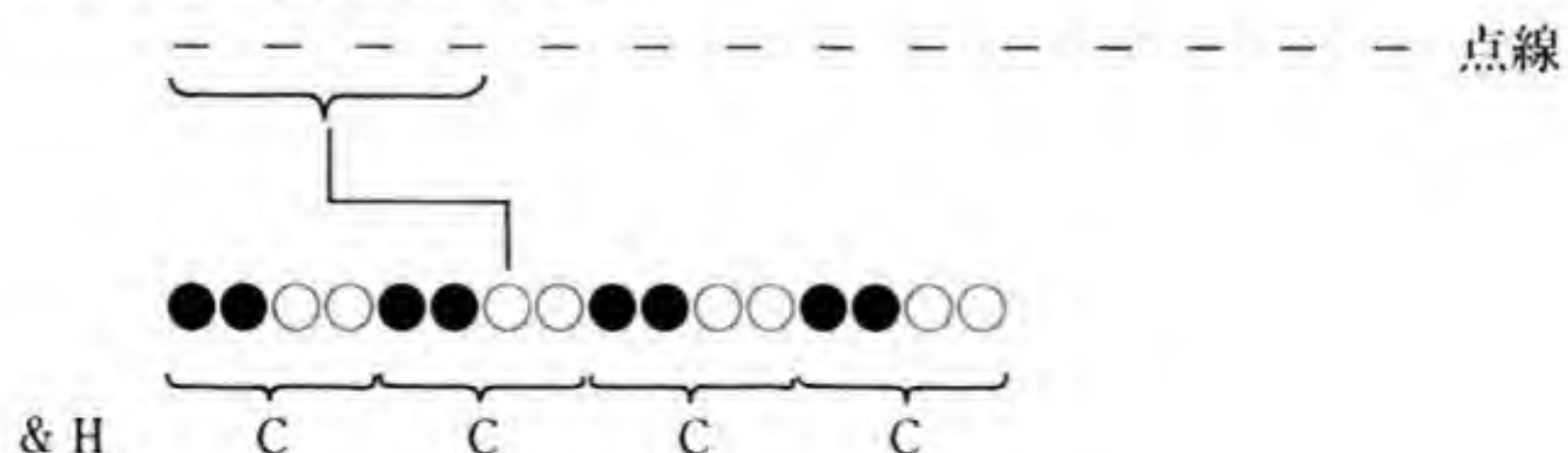
c：パレットコード　　　　ｌｓ：ラインスタイル

ただ単に実線を引くだけならば、入門編「２章．絵を描く、色を塗る」で説明したようにたとえば次の命令を実行するだけでＯＫです。

```
LINE (0, 0) - (319, 199), PSET
```

これで、２点（０，０）、（３１９，１９９）を結ぶ実線が引かれます。線の色は、パレットコードを省略しているため、テキスト画面の文字の色に対応したパレットコードが与えられます。（初期状態では白色です）また、始点座標が省略されると、直前に実行したＬＩＮＥステートメントの終点座標が新たに始点座標としてみなされます。

第３パラメータ（ｍｏｄｅ）には、ＰＳＥＴ，ＰＲＥＳＥＴ，ＸＯＲの３通りの指定が可能です。ＰＳＥＴ，ＰＲＥＳＥＴについては「点を描く」で説明した通りです。ＸＯＲを指定すると、他のグラフィック図形と重なった直線部分の色が反転します。たとえば、すでに白（青＋赤＋緑）でペイントされている所へ、ＸＯＲを使って青のラインを引くと、重なった部分のラインの色は黄（赤＋緑）になります。このパラメータが省略されると、直前に設定したｍｏｄｅの影響をうけます。

第５パラメータ（ｌｓ）には、ラインスタイルを指定します。ラインスタイルとは、点線、破線などを１６ビットのビットパターンで表わしたものです。たとえば、次のような点線のラインスタイルは＆ＨＣＣＣＣで表わされます。

このように、ラインスタイルは、16ドットの範囲内で任意のくり返しパターンを設定することができます。このパラメータを省略すると、実線（ラインスタイル＝&HFFFF）で引かれます。以下に主なラインスタイルとその破線のパターンを示します。

ラインスタイル	破線のパターン
$HFFFF	―――――――――― 実線
$HFF00	― ― ― ― ―
$HCCCC	-------------------- 点線
$HAAAA	....
$HFFCC	――-――――-― 一点鎖線
$HE4E4	――-――-――-―
$HFCCC	――--――--―― 二点鎖線
$HFF24	――-.――-.――

では、実際に直線を引いてみましょう。まず、画面を初期化して、グラフィック画面をクリアします。

INIT ⏎
CLS0 ⏎

次に、2点（0, 0）、（190, 190）を結ぶ1点鎖線を黄色で引きます。

LINE (0, 0) - (190, 190), PSET, 6, &HFFCC ⏎

どうですか？うまく引かれましたか？もし、思いどおりの直線がひかれなければ CTRL + D を押してみてください。さて、正しく表示されたら、今度は同じ座標間をXORモードを使って黄色の実線で結びます。

LINE (0, 0) - (190, 190), XOR, 6 ⏎

どうなりましたか？今まで1点鎖線が表示されていたところが透明になり、今まで透明だった部分に黄色が表示されましたね。このように、XORを指定すると、重なった部分のXOR論理がとられ、色が反転してしまいます。

サンプルプログラム

各種ラインスタイルを実際に表示させてみます。

```
10 'LINE STYLE
20 WIDTH 40,25,0:CLS0:INIT
25 PRINT"ラインスタイル";SPACE$(10);"パターン"
30 RESTORE 100
40 FOR I=2 TO 9
50 LOCATE 0,I*2
60 READ A$
70 PRINT A$
80 LINE(60,I*16+4)-(319,I*16+4),PSET,7,VAL(A$)
90 NEXT
100 DATA &HFFF,&HFF00,&HCCC,&HAAA,&HFFCC,&HE4E4,&HFCCC,&HFF24
```

4.2 長方形を描く

長方形を描く場合、ＬＩＮＥステートメントのパラメータは次のようになります。

```
LINE [(x1, y1)] - (x2, y2), mode [, c], B [, ls]
```

Ｂ：長方形を描く指定

指定した２点間の対角線を引くか、長方形を描くかは、記号"Ｂ"（ＢＯＸの略）があるかないかによって決ります。他のパラメータは、線を引く場合とまったく同じです。

では、実際に長方形を描いてみましょう。まず、画面の初期化とクリアを行ないます。

```
INIT ↵
CLS0 ↵
```

次に、２点（０，０）（１９０，１９０）を対角とする長方形をシアンで描きます。

```
LINE (0, 0) - (190, 190), PSET, 5, B↵
```

さらに、その上から、赤の点線でなぞってみます。

```
LINE (0, 0) - (190, 190), PSET, 2, B, &HCCCC↵
```

これで、赤とシアンの点線で長方形が描かれました。

4.3 長方形を塗りつぶす

長方形を塗りつぶす場合、ＬＩＮＥステートメントのパラメータは次のようになります。

```
LINE [(x1, y1)] - (x2, y2), mode [, c], BF
```

ｃ：パレットコードまたは中間色コード

ＢＦ：長方形を塗りつぶす指定

長方形を描いてその中を塗りつぶすには、記号"ＢＦ"（Ｂｏｘ Ｆｉｌｌの略）を指定します。ただし、この場合、指定できる色はパレットコードだけでなく、中間色コードも指定できます。

たとえば、２点（０，０）、（１９０，１９０）を対角とする長方形をシアンと赤の中間色コードで塗りつぶす場合は、次のステートメントを実行します。

```
LINE (0, 0) - (190, 190), PSET, &H25, BF↵
```

ただし、ＢＦを指定する場合は、ラインスタイルを指定することはできません。そのかわり、**タイリングパターン**を設定することができます。タイリングパターンというのは、より複雑な中間色や色模様を出すための文字列データで、最大８×８ドットの色パターンを定義することができます。最小のタイリングパターンは横８ドットの色パターンで、これを指定するには青、赤、緑各１バイトずつの計３バイトの文字列データが必要です。したがって、８×８のタイリングパターンを定義する場合は、２４バイトの文字列データが必要です。このように、中間色が２色のドットを交互に配置するだけなのに対して、タイリングパターンは、横８ドットの色パターンを自由に設定することができます。タイリングパターンを設定すると、その色パターンを使って長方形を描き、かつ塗りつぶします。タイリングパターンを使用する時、ＬＩＮＥステートメントは次のようになります。

```
LINE [(x1, y1)] - (x2, y2), mode, BF, t$
```

ｔ＄：タイリングパターン（文字列）

タイリングパターンの文字列は、CHR$やHEXCHR$関数によって用意するのが一般的です。

たとえば、横8ドットのパターンを、1ドット1色ずつにして8色をすべて使って定義する場合は次のようにします。

横8ドットのパターン	黒	青	赤	マゼンタ	緑	シアン	黄	白	
	↓	↓	↓	↓	↓	↓	↓	↓	
青データ	0	1	0	1	0	1	0	1	=&H55
赤データ	0	0	1	1	0	0	1	1	=&H33
緑データ	0	0	0	0	1	1	1	1	=&H0F

タイリングパターンt$は次のように表わされます。

t$=HEXCHR$("55330F")

そのタイリングパターンで、2点(0, 0)、(190, 190)を対角とする長方形を塗りつぶすには、次のステートメントを実行します。

```
LINE (0, 0) - (190, 190), PSET, BF, HEXCHR$ ("55330F") ⏎
```

4.4 折れ線を描く

LINEステートメントは、グラフ等に必要な折れ線を簡単に描くことができます。その場合のパラメータは次のようになります。

```
LINE [(x1, y1)] - (x2, y2), (x, y)……- (xn, yn)
```

x_1, y_1：折れ線の始点座標

x_i, y_i：折れ線の頂点座標 (i=2, 3……………n−1)

x_n, y_n：折れ線の終点座標

このように、座標を連続して入力すると、各座標間を結んだ折れ線を描きます。これを利用すれば、任意の形の多角形を簡単に描くこともできます。たとえば、4点(200, 0), (100, 100), (0, 100), (100, 0)を頂点とする平行四辺形を描くには、次のように入力します。

```
LINE (200, 0) - (100, 100) - (0, 100) - (100, 0) - (200, 0) ⏎
```

5 正多角形を描く

POLYステートメント

正多角形に限り、POLYステートメントを使って簡単に描くことができます。

```
POLY (x, y), r [, c, Δθ, θs, θe]
```

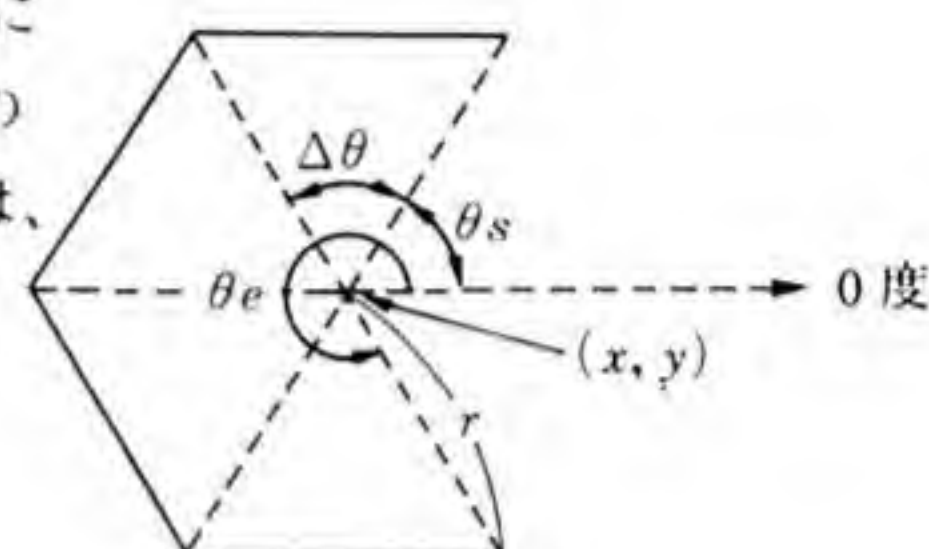
x, y：正多角形の中心座標（ユーザー座標系）

r：中心から頂点までの距離

c：正多角形の辺の色（パレットコード）

Δ θ：ステップ角（０～３６０度）
θ s ：初期角（０～３６０度）
θ e ：終了角（０～３６０度）

多角形は、前述のＬＩＮＥステートメントでも描くことができますが、ＬＩＮＥの場合は各頂点の座標を計算して入力しなければなりません。しかし、ＰＯＬＹステートメントでは、円を描くのと同じ要領で入力することができますので、正多角形の中心（x, y）と半径r等の設定だけで簡単に描くことができます。(円については「6. 円を描く」で述べます)。各パラメータについて以下に示します。

ステップ角（Δ θ）は、正多角形の頂点間の角度で、０度に近いほど真円に近づきます（ただし、０度を指定するとただの線分になってしまいます)。初期値（θ s）と終了角（θ e）は、x座標の増分方向（画面の右方向）を０度とした場合の、多角形を描き始める角度と、描き終える角度を示します。

したがって、一般的に、正n角形を描くには、次のように指定します。

Δ θ＝３６０／n
θ e＝θ s＋３６０

では、実際に、正多角形を描いてみます。まず、画面を初期化してクリアしてください。

ＩＮＩＴ↵
ＣＬＳ０↵

今度は、右側半分だけの半円を描いてみます。右側だけの半円ですから、初期角は９０度、終了角は２７０度で指定します。

POLY (200, 100), 80, 6, 1, 90, 270↵

これで黄色の半円が描かれました。

(注) ＰＯＬＹおよびＣＩＲＣＬＥステートメントにおける頂点距離または半径は、画面モードによって、実際の座標と対応していない場合があります。詳しくは、『ＢＡＳＩＣリファレンスマニュアル』の２.８.７ＣＩＲＣＬＥステートメントの項を参照してください。

サンプルプログラム

ＰＯＬＹで多角形を重ねます。

```
10 INIT:CLS4:WIDTH 40,25,0,0
20 RESTORE 160
30 FOR KAI=1 TO 5
40 READ STE1,STA1,OWA1,STE2,STA2,OWA2
50 C=INT(RND*6)+1
60 FOR X=1 TO 14
70 FOR Y=1 TO 9
80 POLY(X*20,Y*20),20,C,STE1,STA1,OWA1
90 POLY(X*20+2,Y*20),20,C+1,STE2,STA2,OWA2
100 NEXT Y
110 NEXT X
120 PAUSE 20
130 CLS4
140 NEXT KAI
150 GOTO 20
160 DATA 120,0,360,120,90,450
170 DATA 90,0,360,90,45,405
180 DATA 72,0,360,72,90,450
190 DATA 60,0,360,60,30,390
200 DATA 144,0,720,144,90,810
```

サンプルプログラム

時計を描きます。

```
10 INIT:CLS4:WIDTH 40,25,0,0
20 POLY(160,100),80,7,30
30 T$=TIME$
40 IF T$=T1$ THEN 30
50 X=VAL(LEFT$(T$,2))
60 Y=VAL(MID$(T$,4,2))
70 Z=VAL(RIGHT$(T$,2))
80 POLY(160,100),40,0,0,90-X1*30-INT(Y1/2)
90 POLY(160,100),50,0,0,90-Y1*6
100 POLY(160,100),60,0,0,90-Z1*6
110 POLY(160,100),40,5,0,90-X*30-INT(Y/2)
120 POLY(160,100),50,4,0,90-Y*6
130 POLY(160,100),60,6,0,90-Z*6
140 LOCATE 16,16:PRINT T$
150 X1=X:Y1=Y:Z1=Z:T1$=T$
160 GOTO 30
```

6 円を描く

ＣＩＲＣＬＥステートメント

円、楕円および円弧を描くステートメントは、ＣＩＲＣＬＥステートメントです。

```
CIRCLE 〔@〕 (x, y), r〔, c, f, θs, θe〕
```

f：偏平率（＝たて径／横径）

このように、ＣＩＲＣＬＥステートメントのパラメータは、ＰＯＬＹステートメントと似ています。

違いは、ＰＯＬＹステートメントのステップ角が、ＣＩＲＣＬＥステートメントでは偏平率に変わっている点だけです。また、ＣＩＲＣＬＥステートメントでは、@記号がつくかつかないかで、半径及び偏平率の扱いが異なってきます。

(1) @記号がつかない場合

画面に表示される円の大きさは、３２０×２００ドットモードの時を基準にしています。したがって、半径が同じならば、どの画面モードで描いても、画面に表示される円の大きさは同じになります。このことは画面上の座標と半径の大きさとが、必ずしも一致しないことを示しています。偏平率を使用した場合も同様です。偏平率を省略すると、１が設定されます。

(2) @記号がつく場合

画面に表示される円の大きさは、各画面モードの座標に従います。したがって、同じ半径でも画面モードによって円の大きさは異なってきます。偏平率を指定した場合でも同様です。したがって、偏平率を１に設定したからといって、画面モードによっては真円にならない場合があります。偏平率を省略すると、画面上に真円が描かれるように、自動的に偏平率を設定します。

なお、詳しくは、『ＢＡＳＩＣリファレンスマニュアル』の２．８．７ＣＩＲＣＬＥステートメントおよび２．８．８ＣＩＲＣＬＥ@ステートメントの項を参照してください。

サンプルプログラム

専用ディスプレイテレビにおいて、ＣＩＲＣＬＥステートメントとＣＩＲＣＬＥ@ステートメントにおける半径の取扱いを示すプログラムです。

```
10 'CIRCLE
20 INIT:CLS4:OPTIONSCREEN0
30 G=0:RES=1:GOSUB 70
40 G=0:RES=2:GOSUB 70
50 G=1:RES=2:GOSUB 70
60 END
70 'SUB
80 FOR X=40 TO 80 STEP 40
90 WIDTH X,25,G,RES
100 PRINT "WIDTH ";X",25,";G;",";RES
110 PRINT "   RED.....CIRCLE (200,100),100,2"
120 PRINT "   GREEN...CIRCLE@(200,100),100,4"
130 CIRCLE(200,100),100,2
140 CIRCLE@(200,100),100,4
150 A$=INKEY$:IF A$=""THEN 150
160 NEXT
170 RETURN
```

7 色を塗る(中間色の指定)

PAINTステートメント

グラフィック曲線で囲まれた任意の形の図形を塗るには、PAINTステートメントを使います。ここでは中間色を指定します。
本機では、赤、青、緑の3色を1ドット単位に自由な組合わせで基本の8色以外の中間色を作ることができます。これを**タイリングペイント**といいます。

〔1〕 PAINT (x, y), c, b〔b_1, b_2, ……, b_6〕
〔2〕 PAINT (x, y), t\$, b〔, b_1, b_2, ……, b_6〕

c：パレットコードまたは中間色コード
t\$……タイリングパターン
b, b_1, b_2, ……, b_6：境界色（パレットコード）

〔1〕の書式で、中間色を指定するときcを次のように表わします。

&Hmn（またはm*16+n）
m＝0～7のパレットコード
n＝0～Fのコード

こうすれば、mとnの2色を混合した中間色を出すことができます。
（中間色一覧表は「2章・5.中間色コード」を参照してください。）
赤色を描いてその中を中間色（紫）で塗ります。

```
10 WIDTH 40,25,0:INIT
20 CLS4
30 CIRCLE(200,100),80,2
40 PAINT(200,100),&H12,2
```

〔2〕の書式で、中間色を指定するときは、タイリングパターンt\$はCHR\$やHEXCHR\$関数によって表わします。
たとえば、黄緑色は黄色と緑色を1ドット単位で交互に点灯することにより表示できます。

CHR\$（&H00）+CHR\$（&HAA）+CHR\$（&HFF）…黄緑色
またはHEXCHR\$（"00AAFF"）

青→	0	0	0	0	0	0	0	0	→&H00
赤→	1	0	1	0	1	0	1	0	→&HAA
青→	1	1	1	1	1	1	1	1	→&HFF
	黄	緑	黄	緑	黄	緑	黄	緑	

このデータをPAINT文に入れますと

PAINT (x, y), CHR\$ (&H00) +CHR\$ (&HAA) +CHR\$ (&HFF)

または

PAINT (x, y), HEXCHR\$ ("00AAFF")

8　色を瞬時に変える

ＰＡＬＥＴステートメント

ＰＡＬＥＴステートメントは、グラフィック画面上の色を瞬時に他の色に変換します。

```
PALET p, c, b
```

p：パレットコード（0～7）
c：カラーコード（0～8）
b：ボーダーカラー（0または1）

ＰＡＬＥＴステートメントは、色の変換を行ないますが、実際に塗りつぶして変換するわけではありません。したがって、瞬時の色変換が可能です。変換は、パレットコードをどのカラーコードに対応させるか、で行ないます。パレットコードおよびカラーコードについては「2章　ディスプレイモード」の項を参照してください。

たとえば、パレットコード1は、初期状態ではカラーコード1すなわち青色に対応していますが、赤色（カラーコード2）に変換するには、次の命令を実行します。

```
PALET1, 2⏎
```

また、ＰＡＬＥＴステートメントによりグラフィックの透明と青の2色を、黒色に変換することができます。黒色は、カラーコード8に割り当てられており、ＰＡＬＥＴ0，8またはＰＡＬＥＴ1，8のどちらかを指定します。前者が透明を黒に、後者が青を黒に変換します。ただし、黒色といっても、コンピュータモードでは透明とまったく変わりません。透明との違いがでるのはスーパーインポーズ時のみです。透明はバックのテレビ画面を透き通しますが、黒色は透き通しません。

また、第3パラメータが1の時、画面のボーダー部分を黒色にします。0の時は透明のままです。これによって、スーパーインポーズ時にテレビ画面を黒ぬきにすることができます。

また、ＰＡＬＥＴステートメントは次のように設定することもできます。

```
PALET@c1, c2, c3, c4, c5, c6, c7, c8
```

c_1, c_2, ………………… c_8：カラーコード

この指定方法は、複数のパレットコードを一度に変換する場合に用いられます。パレットコードは、パラメータの位置に対応しています。すなわち、第1パラメータがパレットコード0に、第2パラメータがパレットコードの1に、第8パラメータがパレットコード7にそれぞれ対応しています。たとえば、パレットコード1を赤に、パレットコード7を緑に変換するためには、次の命令を実行します。

```
PALET@0, 2, 2, 3, 4, 5, 6, 4⏎
```

サンプル
プログラム

花火の点滅する感じをお楽しみください。

```
10 '花火
20 WIDTH 40,25,0:CLS4:INIT
30 SCREEN1,1:KLIST0:PRINT "Wait a moment!"
40 DEFFNX(CX)=COS(RAD(CX))*CR
50 DEFFNY(CY)=SIN(RAD(CY))*CR
60 CCX=INT(RND*300)+10
70 CCY=INT(RND*150)+10
80 H=INT(RND*10)+6
90 SCREEN 1,0:KLIST0
100 FOR CR=1 TO H*7 STEP H
110 FOR DEG=0 TO 360 STEP H
120 C=INT(RND*7)+1
130 PSET(FNX(DEG)+CCX,FNY(DEG)+CCY,C)
140 NEXT DEG
150 NEXT CR
160 SCREEN 1,1:CLS4
170 SOUND 7,&HFE:PLAY"-E1"
180 FOR UP=200 TO CCY STEP -1
190 C=INT(RND*7)+1
200 PSET(CCX,UP,C):PSET(CCX+1,UP,C)
210 PRESET(CCX,UP,C):PRESET(CCX+1,UP,C)
220 NEXT UP
230 SCREEN0,0
240 SOUND 7,&HF7:SOUND 6,43:SOUND 8,16:SOUND 11,0:SOUND 12,150:SOUND 13,0
250 FOR I=1 TO 15
260 FOR J=1 TO 7
270 PALET J,((J+1)MOD 7)+1
280 NEXT J
290 NEXT I
300 FOR I=1 TO 7
310 PALET I,0
320 PAUSE 1
330 NEXT I
```

9 テキスト文字とグラフィック画面の優先順位を決める

PRWステートメント

PRWステートメントは、テキスト画面上の文字とグラフィック画面上の図形が重なった時、どちらを優先して表示させるかを決めるステートメントです。

```
PRW 〔n〕
```

n＝0～255

グラフィック画面の各色とテキスト文字との優先順位は、nの値で決定されます。nの値はコンピュータ内部では8ビットで表現されており、各ビットは次のように各パレットコードに対応しています。

	7	6	5	4	3	2	1	0	
n：	白	黄	シアン	緑	マゼンタ	赤	青	透明	（初期状態のパレットコード）

最下位ビットが透明に対応し、最上位ビットが白に対応しています。この時、あるビットの内容が1に設定されると、その色をもつグラフィックが優先され、0に設定されるとテキスト文字が優先されて表示されます。たとえば、グラフィックの青と緑をテキストより優先させて表示させるには、次のように設定します。

PRW &B00010010⏎ (=PRW 18)

ＰＲＷステートメントは、パレットコードに対する命令ですので、たとえば、ＰＡＬＥＴ１，２を実行してパレットコード１を赤に変換すると、その部分の赤がテキストより優先されます。ただし、パレットコード２の赤はテキストに隠れたままです。このように、ＰＲＷステートメントを利用すると、見かけ上同じ色でもテキストとの優先順位を変えることができます。

優先順位を初期状態にもどすには、ＰＲＷ⏎を実行します。

10 文字パターンの描画

SYMBOLステートメント

ＳＹＭＢＯＬステートメントは､グラフィック画面上に文字列を指定の角度とサイズで描きます。

SYMBOL (x, y), x$, h, v, { c' / t$ }, θ, mode

x, y ： 文字列の表示を開始する左上のドットの座標（画面座標系）
x＄ ： 表示する文字列
h ： 横方向の倍率（１、２、３……）
v ： 縦方向の倍率（１、２、３……）
c' ： 文字の色（パレットコードまたは中間色コード）
θ ： 描く方向の指定（０～３）
mode ：ＰＳＥＴ，ＰＲＥＳＥＴ，ＸＯＲ，文字式または" "
t＄ ： タイリングパターン

h, v は、それぞれ横、縦方向の倍率を指定します。設定値はすべて整数とみなされます。小数は四捨五入されます。この値によっては文字が画面から、はみ出してしまうので注意してください。

t＄はタイリングパターンで、ＨＥＸＣＨＲ＄などで指定します。タイリングパターンについては「７.色を塗る」の項を参照してください。

θは、表示する文字の方向を示しています。

θ＝０では、通常表示
θ＝１では、９０度左に回転したもの
θ＝２では、１８０度左へ回転したもの
θ＝３では、２７０度左へ回転したもの
となります。

ｍｏｄｅは、文字列の表示モードを示し、ＰＳＥＴ，ＰＲＥＳＥＴ，ＸＯＲを指定するとグラフィック画面に表示し、文字式または""ヌルコードを指定するとテキスト画面に表示します。次に例を示します。

```
SYMBOL (100, 100), "A", 1, 1, 1, 0, PSET⏎
```

これはグラフィック画面に、青色のＡという文字をたて横１倍で表示します。

```
SYMBOL (100, 100), "B", 2, 2, 4, 1, PSET⏎
```

これら、グラフック画面に緑色のＢという文字をたて横２倍の大きさで表示します。その時Ｂは９０度左に回転されて表示します。

```
SYMBOL (0, 0), "A", 1, 1, 1, 0, "B"
```

これは、テキスト画面にキャラクタＢでＡという文字を表示します。

１１　ＧＥＴ＠とＰＵＴ＠

ＧＥＴ＠はグラフィック画面の情報を配列変数に読み込むステートメントです。

また、ＰＵＴ＠というステートメントはＧＥＴ＠の逆で、配列変数の情報を画面に描くステートメントです。

通常、これらの２つのステートメントを組み合わせて用いることによって、グラフィック画面のデータを別の位置に移動させることができます。

これらのステートメントは、画面データを一度配列変数に入れるので、データが入るだけの配列変数領域をあらかじめ定義しておく必要があります。

たとえば、次の例では、画面左上に描いた円を赤でぬり、それと同じ円を別の場所に描いています。

```
10 INIT:CLS4
20 DIM A%(100)
30 CIRCLE(10,10),5:PAINT(10,10),2,7
40 GET@(0,0)-(20,20),A%,7
50 FOR I=0 TO 100 STEP 20
60 PUT@(I,I)-(I+20,I+20),A%,PSET,7
70 NEXT
```

ＧＥＴ＠およびＰＵＴ＠の書式は

```
GET@(x1, y1) - (x2, y2), 配列名 ,パレットコード
PUT@(x1, y1) - (x2, y2), 配列名 { ,PSET ,パレットコード }
                                 { ,PRESET            }
                                 { ,XOR               }
                                 { ,OR                }
                                 { ,AND               }
                                 { ,NOT               }
```

x_1、y_1：読みこむ領域の左上隅の座標（画面座標系）

x_2、y_2：読みこむ領域の右下隅の座標（画面座標系）

ＰＵＴ＠ステートメントでは表示のときにＰＳＥＴ，ＰＲＥＳＥＴ，ＸＯＲ，ＯＲ，ＡＮＤ，ＮＯＴとモードがあり、画面にすでにあるドットと配列中のドットの演算結果を画面に表示することもできます。また、パレットコードを省略するとテキスト画面に対するステートメントになります。

12 グラフィックパターンの描画

POSITION・PATTERNステートメント

まず、グラフィック画面にドットパターンを描く際には描画開始位置を指定します。

それにはＰＯＳＩＴＯＮステートメントを用意します。

```
POSITION x, y
```

x, y：ドットパターン表示の左上隅の座標（画面座標系）

ＰＯＳＩＴＩＯＮステートメントの実行により描画開始位置を指定した後、ＰＡＴＴＥＲＮステートメントでドットパターンを描画します。

```
PATTERN n, x$ 〔, y$………〕
```

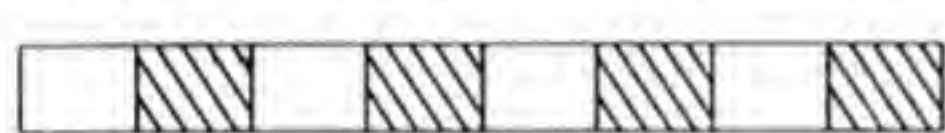
n：ドットパターンのたて方向の段数を指定し、n＜0のとき下の方向に重なりn＞0のとき上の方向に重なります。

x$, y$：ドットパターンを表わす文字式。

たとえば

POSITION10, 10

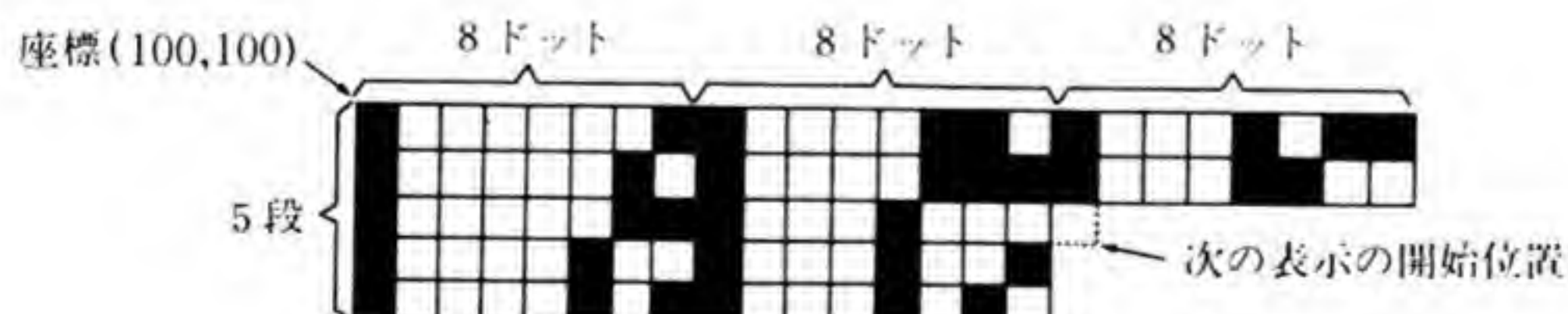
PATTERN−1, CHR$ (&H55)

と入力するとx座標10, y座標10を描画開始位置として、ドットパターン（▧がセット。&H55＝&B01010101）が表示されます。

また、ドットパターンの積み重ねは段数を指定するnの値より、積み重ねの方向と段数が決められます。nが負のとき上から下へ、nが正のとき下から上へ積み重ねられます。

表示色は、テキスト画面の文字の色に対応したパレットコードで描れます。

たとえば

```
10 WIDTH 40:INIT
100 A$=HEXCHR$("8182838485868788898A8B8C")
110 POSITION 100,100
120 PATTERN-5,A$
```

を実行すると下の様に表示されます。

5章
日本語処理

入門編「漢字を表示する」では漢字ＢＡＳＩＣを利用して、漢字、ひらがな等を表示してみました。この漢字ＢＡＳＩＣ（ＣＺ－８ＦＢ０２）は、漢字、ひらがな等の文字をＢＡＳＩＣプログラム上で容易に扱えるように、カナ-漢字変換機能や外字登録機能を持っています。
この章では、これら全角文字と呼ばれる文字の扱いについて説明します。

1　全角文字とは

全角文字とは、漢字やひらがなのように、そのフォントが１６×１６ドットで構成された文字のことを指します。それに対して、そのフォントが８×８または１６×８ドットで構成された文字を、**半角文字**と呼んでいます。

従来のコンピュータでは、通常、英数字・カタカナ・記号・セミグラフィック等の半角文字のみ扱っており、漢字・ひらがな等の全角文字はプログラム中ではコード（ＪＩＳ漢字コード、区点コード等）として扱うことしかできませんでした。（たとえば、"亜"という漢字はＫＡＮＪＩ＄（１６０１）で表わします）しかし、**本機の漢字ＢＡＳＩＣでは、これらの全角文字をプログラム中で半角文字と同様に文字の形で扱うことができます。**それは、次のような場合において可能です。

(1) ＰＲＩＮＴ文等のダブルクォーテーション（"）で囲まれた部分
(2) ファイル名及びディレクトリ名
(3) ＲＥＭ文
(4) ＤＡＴＡ文
(5) 文字変数

このように、本機の漢字ＢＡＳＩＣでは、プログラム中に漢字やひらがなを混入させることができますが、ＢＡＳＩＣ内部ではこれらの全角文字は２バイトのコードとして処理されています。この意味から、従来の文字（半角文字）が**１バイトコード文字**と呼ばれているのに対して、全角文字は**２バイトコード文字**とも呼ばれています。

したがって全角文字の１文字分は半角文字の２文字分とみなされますので、たとえば、ファイル名の長さ等、文字数に注意する必要があります。また、本機の漢字ＢＡＳＩＣでは、全角文字を扱う方式として、従来の半角文字のセミグラフィック部分の１バイトコードを利用するＳＨＩＦＴ ＪＩＳコード体系をとっています。したがって、全角文字を扱う場合はセミグラフィック文字を混在させることができません。（ただし、英数字、カタカナ、記号等は使用できます）全角文字を表示させるか否かは、次のＢＡＳＩＣコマンドによって決められます。

KMODE　0………半角文字のみ扱うモード（セミグラフィック文字使用可）
KMODE　1………全角文字も扱うことができるモード（セミグラフィック文字使用不可）

また、全角文字の場合その文字のフォントがたて１６ドットで構成されていますので、表示画面が標準ディスプレイモードの時には、全角文字は次の画面モードの時のみ表示することができます。

ＷＩＤＴＨ　８０，１２，０，１
　　　　　　　　　　　　　０

ＷＩＤＴＨ　４０，１２，０，１
　　　　　　　　　　　　　０

ＷＩＤＴＨ　８０，１０，０，１
　　　　　　　　　　　　　０

ＷＩＤＴＨ　４０，１０，０，１
　　　　　　　　　　　　　０

2　全角文字の入力方式

(1)ひらがな、カタカナ、英数字等の全角文字の入力方式

次の３通りの方式があります。また、これらの方式では半角文字の入力も可能です。

①**英数字直接入力方式**－アルファベット、数字等の入力を簡単に行なえる.

②**ローマ字ーカナ変換方式**－ローマ字表記にて入力し、それをひらがなまたはカタカナに変換する。

③**カナ入力方式**－［カナ］キーをロックしてカナ表記にて入力し、それをひらがなまたはカタカナに変換する。

(2)漢字への変換機能

次の２通りの方式があります。

①**コード入力方式**－漢字のコード（ＪＩＳ漢字コードまたは区点コード）を入力して、直接、漢字、ひらがな等の全角文字に変換する。

②**カナー漢字変換方式**－(1)の②ローマ字ーカナ変換方式および③カナ入力方式において入力した、ひらがなまたはカタカナを漢字に変換する。

この方式には次の方式があります。

- **一字変換方式**…ひらがなまたはカタカナの文字列の先頭の１文字のみを判断して、その読みを持つ漢字をＪＩＳ漢字コードの配列通りに羅列して、変換する。
- **音訓変換方式**…漢字の音読み、訓読みを入力し変換する。

なお、一字変換方式は、コンピュータの内部プログラムとして存在しますのでディスク等は一切不要ですが、音訓変換方式は、システムディスクの中の音訓辞書が必要です。

(3)日本語入力モードに入るための条件

〔条件１〕

標準ディスプレイモードの場合、画面モードを次のいずれかに設定すること。（高解像度をディスプレイモードの場合は、制限ありません）

○ＷＩＤＴＨ　８０，１２，０，１
　　　　　　　　　　　　　　０

○ＷＩＤＴＨ　４０，１２，０，１
　　　　　　　　　　　　　　０

○ＷＩＤＴＨ　８０，１０，０，１
　　　　　　　　　　　　　　０

○ＷＩＤＴＨ　４０，１０，０，１
　　　　　　　　　　　　　　０

〔条件2〕

ＣＯＮＳＯＬＥステートメントによって、表示画面の最下行が変換フィールドエリアとして確保されていること。

（例）

○２５行モードならば、ＣＯＮＳＯＬＥ　０，２４
○１２行モードならば、ＣＯＮＳＯＬＥ　０，１１
○２０行モードならば、ＣＯＮＳＯＬＥ　０，１９
○１０行モードならば、ＣＯＮＳＯＬＥ　０，９

ただし、ＣＯＮＳＯＬＥステートメントだけでなく、ＩＮＩＴ命令や CTRL ＋ D キー入力によっても、変換フィールドエリアは確保されます。

〔条件3〕

"ＫＭＯＤＥ　１"を実行してあること。

以上の条件が満たされているならば、

（a） CTRL キーを押しながら XFER キーを押す。（CTRL ＋ XFER）

または

（b） SHIFT キーを押しながら XFER キーを押す。（SHIFT ＋ XFER）

と、日本語入力モードに、はいることができます。

なお、ＢＡＳＩＣの起動時では、上記の条件〔１〕～〔３〕までを満たしています。

日本語入力モードに入ると、表示画面の最下行に変換フィールドが現われます。

全角文字への変換は、すべてこの変換フィールド内で行なわれます。また、日本語入力モードから通常の状態に戻るには、再び前記(a)または(b)を実行します。

(a)40桁モードの場合の入力画面　　(b)80桁モードの場合の入力画面

日本語入力モードにおける表示画面例

〔注〕日本語入力モードにおける表示画面は、４０桁と８０桁モードとでは若干異なりますが、動作には変わりはありません。

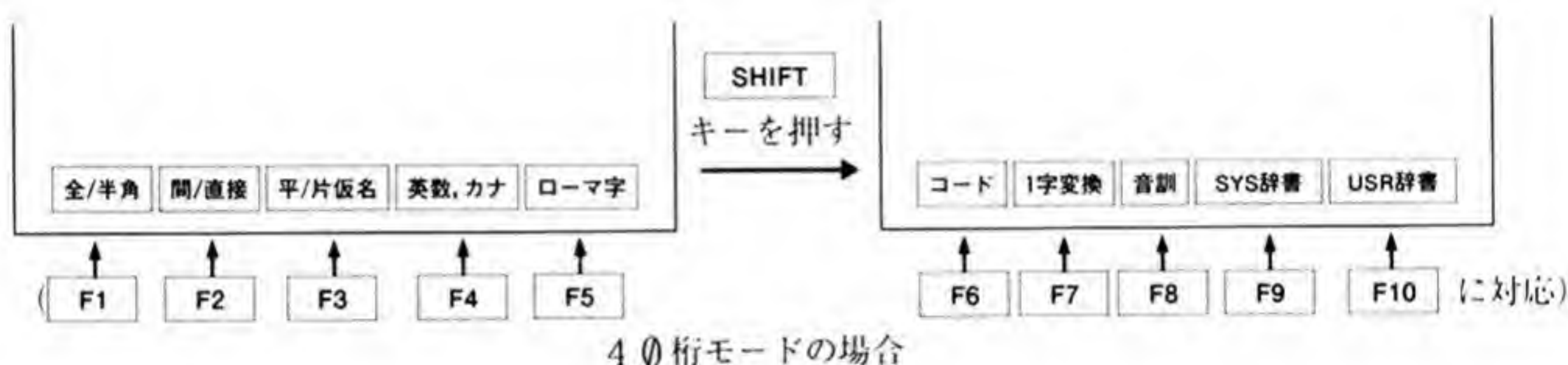
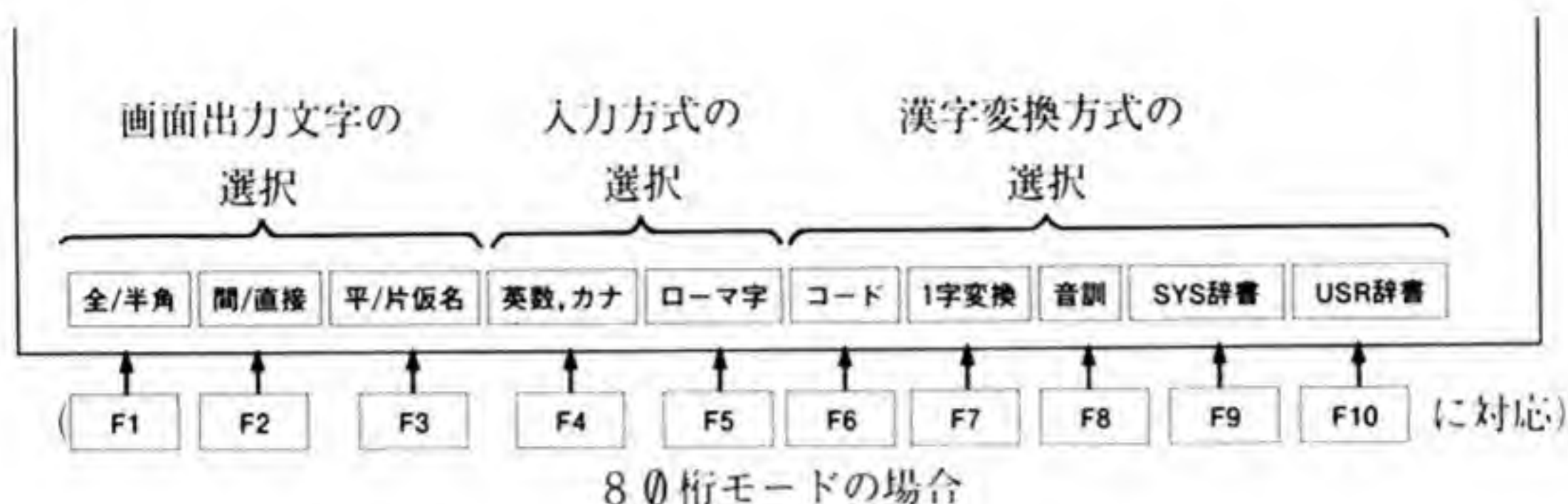
3　入力モードの選択

日本語入力モードでは、画面に出力される文字、入力方式、漢字変換方式をファンクションキー F1 ～ F10 によって選択することができます。

3.1 HELP キーについて

SHIFT + XFER または CTRL + XFER で日本語入力モードにし、次に HELP キーを押してください。すると最下行の変換フィールドは次のように変わります。

8 0 桁モードの場合

4 0 桁モードの場合

このように、入力モードの確認は HELP キーによって行ないます。上記の画面で、**現在のモードは赤字**で表示されます。(各モードについては次の3.2で説明します) これらのリストを表示したままの状態で、F1 ~ F10 キーを押せば、入力モードの変更ができます。日本語入力モードに戻るためには、再び HELP キーを押します。

3.2 各入力モードの選択

(1)画面出力文字の選択 (F1, F2, F3 について各々選択)

F1 ：全角／半角…………全角文字か半角文字のどちらを表示させるかを選択します。

F2 ：直接／間接…………直接、テキストエリアのカーソル位置に出力するか、またはいったん変換フィールド内に出力するかを選択します。

F3 ：平仮名／片仮名……ローマ字―カナ変換方式またはカナ入力方式で入力した時に、ひらがなを出力するか、カタカナを出力するかを選択します。

(2)入力方式の選択 (F4, F5, カナ の中から1つ選択)

F4 ：英数字直接入力方式……キーボード上のアルファベット、数字、記号の通りに入力します。この入力方式を選択すると、自動的に画面出力文字も英数字、記号となります。

F5 ：ローマ字―カナ変換方式……アルファベットを入力すると、自動的にローマ字とみなされ、ひらがなまたはカタカナに変換されます。

※ カナ キーロックによるカナ入力方式
入力方式が F4 の英数字または F5 のローマ字―カナ変換のモードにある時、
カナ キーをロック（押し下げた状態）すれば、直接カナによる入力ができます。

(3)漢字変換方式の選択（F6，F7，F8，F9，F10 の中から1つ選択）

F6 ：コード入力方式……全角文字のコードを直接入力します。コードにはＪＩＳ漢字コード（"16進"と表示）と区点コード（"区点"と表示）とがあり、このキーを押す毎に切換わります。
（SHIFT ＋ F1）

F7 ：一字変換方式……漢字の読みを1文字だけ識別（他の文字は無視）して漢字に変換します。
（SHIFT ＋ F2）

F8 ：音訓変換方式……漢字の音読み、訓読みを入力して、漢字に変換します。
（SHIFT ＋ F3）

F9 ：システム辞書変換方式…日本語百科ワードパワー（ＷＯＲＤ　ＰＯＷＥＲ）ディスクやシステム辞書ディスク（オプション：ＣＺ-１１１ＳＦ）を使用するときの方式です。
（SHIFT ＋ F4）

F10 ：ユーザー辞書変換方式………ターボ博士レキシコン（ＬＥＸＩＣＯＮ）ディスクやユーザー辞書ディスク（オプション：ＣＺ-１１１ＳＦ）を使用するときの方式です。
（SHIFT ＋ F5）

(4)一字変換、音訓変換、システム辞書・ユーザー辞書変換方式について

①一字変換機能 …コンピュータの内部プログラムとして常駐しています。
②音訓変換方式 …これを利用するためには、音訓辞書が必要です。音訓辞書はシステムディスクの中にあります。
③システム辞書，ユーザー辞書変換方式…それぞれＷＯＲＤ　ＰＯＷＥＲディスク、ＬＥＸＩＣＯＮディスクまたはシステム・ユーザー辞書（オプション：ＣＺ-１１１ＳＦ）が必要です。

なお②、③の変換方式には、変換エリアをメモリ内部に確保するために、ＬＩＭＩＴコマンド（またはＣＬＥＡＲコマンド）でエリアの確保を行なう必要があります。各方式におけるエリア領域は次のとおりです。

ＬＩＭＩＴ＆ＨＦØØØ（または　ＣＬＥＡＲ＆ＨＦØØØ）

※入力モードの選択はファンクションキー F1 ～ F10 のかわりに GRAPH キーを押しながら数字キー 1 ～ 0 を入力することによって行なうことができます。
（ただし、数字キーとしてテンキーを用いることはできません）

〔例〕○全角／半角の切換え方法… F1 または GRAPH ＋ 1 を入力する。
○音訓変換方式への切換え方法… F8 （SHIFT ＋ F3）または GRAPH ＋ 3 を入力する。

4 漢字変換方式について

漢字変換方式には前項でも説明したように、次の５種類があります。

1．コード入力方式
2．一字変換方式
3．音訓変換方式
4．システム辞書変換方式
5．ユーザー辞書変換方式

1,2,3について説明します。4,5については『ターボ博士レキシコン、日本語百科ワードパワーの説明書』またはオプションのシステム・ユーザー辞書(CZ-111SF)の説明書をお読みください。
なお、上記１～５の漢字変換方式を選択するには次のように行なってください。

①ディスクBASIC（CZ−8FBØ2）を起動します。
②次に、日本語入力モードに入ります。
（SHIFT＋XFER または CTRL＋XFER をキー入力します）
③ HELP キーを押すと、

BASIC起動時は音訓変換方式になっています。

- コード入力方式を選択するときは、
- 一字変換方式を選択するときは、
- 音訓変換方式を選択するときは、
- システム辞書変換方式を選択するときは、
  ドライブØにWORD POWERまたはシステム辞書ディスクを入れて
- ユーザー辞書変換方式を選択するときは、
  ドライブØにLEXICONまたはユーザー辞書ディスクを入れて

のキーを押してください。
④次に HELP キーを押すと、日本語入力モードに戻っています。

〔平仮名〕■	全/半角	間/直接	ローマ字	1字変換

一字変換入力方式にした例

4.1 コード入力方式

○コード入力方式を選択するには
F6（SHIFT＋F1キー）または GRAPH＋6キーを入力します。

○コードには　**ＪＩＳ漢字コードと区点コードの２種類があり**どちらで入力するかは同じく

F6（SHIFT＋F1キー）またはGRAPH＋6キー

を入力することによって切換えることができます。

ＪＩＳ漢字コードが選択されると　〔１６進〕

区点コードが選択されると　〔区点〕

の文字が表示されます。

○ＪＩＳ漢字コードと区点コードの間には次式で示すような関係があります。ただしＪＩＳ漢字コードは１６進数、区点コードは１０進数で表わします。

区点コード（上位バイト）＝（ＪＩＳ漢字コード（上位バイト）－２０Ｈ）$_{10}$
区点コード（下位バイト）＝（ＪＩＳ漢字コード（下位バイト）－２０Ｈ）$_{10}$

ＪＩＳ漢字コード（上位バイト）＝（区点コード（上位バイト）＋３２）$_{16}$
ＪＩＳ漢字コード（下位バイト）＝（区点コード（下位バイト）＋３２）$_{16}$

○コード入力方式が選択されると、変換フィールド内には**数字の０～９およびアルファベット大文字のＡ～Ｆの入力以外受けつけられません。**

○コードの入力は**４桁**で行ない（上位２桁：上位バイト、下位２桁：下位バイト）

ＪＩＳ漢字コード選択時には　１６進数

区点コード選択時には　１０進数

とみなされます。

○また、コードと漢字等の全角文字とは１対１に対応しており、その対応関係の詳細は別表に示される通りです。以下にその概略を示します。

JIS漢字コード（16進数）	区点コード（10進数）	対応文字（全角文字）
0021～207E	0001～0094	入力無効（無表示）
2121～2F7E	0101～1594	第１水準　非漢字
3021～4F7E	1601～4794	第１水準　漢字
5021～757E	4801～8594	第２水準　漢字（オプション）
7621～7660	8601～8664	外字（PCG　登録文字）

（注）以下のコードを入力しても無効となります。（＊……任意の数）

ＪＩＳ漢字コード：＊＊００～＊＊２０、＊＊７Ｆ～＊＊ＦＦ

区点コード　　　：＊＊００、＊＊９５～＊＊９９

（注）コード入力方式では、XFERキーを押す必要はありません。４桁のコードを入力した時点で、全角文字に変換されてしまいます。

〔例１〕変換フィールド内でＪＩＳ漢字コード３０２１を入力すると、テキストエリア内に漢字"亜"が表示されます。

〔例２〕変換フィールド内で区点コード１６０１を入力すると、テキストエリア内に漢字"亜"が表示されます。

亜■
〔区点〕■　　　　　　　　　　　全/半角 間/直接 ローマ字 コードIN

○全角文字には、漢字、非漢字文字以外に、**ユーザー登録文字（外字）**があります。
外字はＰＣＧを利用しており、次の２種類があります。
　外字全角文字…ＰＣＧ４キャラクタ分に定義
　外字半角文字…ＰＣＧ２キャラクタ分に定義
（外字の定義方法については『アプリケーションソフトの説明書』の「デフチャーツール」の項を参照してください。）
ここでは**外字全角文字の表示方法**についてのべます。

コード入力方式では、外字全角文字を表示させるために、外字の各文字にコードを割り当てています。外字用のコードは、第１水準および第２水準の全角文字と衝突しないように、次のように割り当てられています。
　ＪＩＳ漢字コード：７６２１～７６６０（６４文字分）
　区点コード　　　：８６０１～８６６４（　〃　）
したがって、外字全角文字を表示させるためには上記コードのいずれかをキー入力します。
上記コードのどれがＰＣＧのどのキャラクタに対応しているかについては、『アプリケーションソフトの説明書』の「デフチャーツール」を参照してください。

〈注意〉
外字半角文字は、コード入力方式では表示できません。外字半角文字を表示するためには次のようにします。
　ＣＧＥＮ２⏎
　ＰＲＩＮＴ＃０，ＣＨＲ＄（Ｘ）⏎
ただし、Ｘは０から＆ＨＦＥまでの偶数のＰＣＧキャラクタコードです。

４．２ 一字変換方式

カナ—漢字変換方式の一つで、変換フィールド内に入力したカナの最初の１文字を判断して、先頭にその読みを持つ漢字を、ＪＩＳ漢字コードの順番通りに羅列します。羅列の仕方は、９文字を一つの漢字グループとみなして１度に最大９文字ずつ、該当する漢字がなくなるまで交代に表示します。

(1)変換例　一字変換方式によって、漢字"日本"を表示させます。
①入力モードを全角文字出力（ F1 ）、間接出力（ F2 ）、平仮名出力（ F3 ）ローマ字入力（ F5 ）、一字変換方式（ F7 ）に設定します。
②変換フィールドに N I C H I 　HTAB　 H O X
の各キーを入力します。

■
〔平仮名〕にちｉほん■　　　←全/半角 間/直接 ローマ字 1字変換

③ XFER キーを押します。すると、"にち"の先頭文字"に"の読みを頭に持つ漢字が羅列されます。さらに、目的の漢字が見つかるまで XFER キーかカーソルコントロールキー⇩かスペースキーを押し続けます。

④目的の漢字"日"を取り出します。(↲キーを押すか、数字キー 1 を押します)

⑤同様に、 XFER キーを押してカナ"ほん"の先頭文字"ほ"の読みを頭にもつ漢字を羅列します。

〔平仮名〕ほん

保舗鋪 圃捕歩 甫補輔
↓ XFER キー入力
穂募墓 慕戊暮 母簿菩
↓ XFER キー入力
倣俸包 呆報奉 宝峰峯
↓ XFER キー入力
崩庖抱 捧放方 朋法泡
↓ XFER キー入力
烹砲縫 胞芳萌 蓬蜂褒
↓ XFER キー入力
訪豊邦 鋒飽鳳 鵬乏亡
↓ XFER キー入力
傍剖坊 妨帽忘 忙房暴
↓ XFER キー入力
望某棒 冒紡肪 膨謀貌
↓ XFER キー入力
貿鉾防 吠頬北 僕卜墨
↓ XFER キー入力
撲朴牧 睦穆釦 勃没殆
↓ XFER キー入力
掘幌奔 本翻凡 盆

日■
〔平仮名〕ほん

⑥目的の漢字"本"を取り出します。(数字キー 4 を押します)

以上で　漢字"日本"が入力されます。

(2)ところで、一字変換方式にはもう1つ別の変換の仕方があります。それは、変換フィールドに入力したキャラクタの先頭の1文字を判断して、そのキャラクタコードをJIS漢字コードの上位バイトに対応させた場合の該当する全角文字を羅列する方式です。たとえば、キャラクタ"！"(キャラクタコード：&H21)を入力した場合、変換フィールドエリアに羅列させる全角文字は、JIS漢字コード2121～217Eまでの94個の文字群になります。この方式ですと、入力文字のキャラクタコードがそのままJIS漢字コードの上位バイトに対応しますので、変換フィールド内に入力するキャラクタは"！"(キャラクタコード：&H21)から"u"(キャラクタコード：&H75)までの85文字ということになります。以下に、入力キャラクタと、対応するJIS漢字コードとの関係を簡単にまとめます。

入力キャラクタ	キャラクタコード	JIS漢字コード	補　　足
！～／	&H21 ～&H2F	2121～2F7E	第1水準　非漢字
0～O	&H30 ～&H4F	3021～4F7E	第1水準　漢字
P～u	&H50 ～&H75	5021～757E	第2水準　漢字(オプション)

羅列された漢字グループから目的の漢字を取り出す方法は、通常の方式とまったく同じです。詳しくは、『入門編』の「5章　漢字を表示する」の項を参照してください。

(注)入力方式としてローマ字―カナ変換方式またはカナ入力方式を選択している場合は、アルファベットや記号の一部を入力することができません。その場合は、英数字直接入力方式に切換えてください。

4.3 音訓変換方式

カナ―漢字変換方式の1つで、変換フィールド内に漢字の音読みまたは訓読みを入力して、それを音訓辞書を利用することによって全角文字に変換します。ディスクBASICを起動させたとき、日本語入力モードにすると音訓変換方式が初期状態として設定されます。

この音訓変換方式の利用については入門編「5章　漢字を表示する」で説明していますので、参照してください。

ところで、音訓変換方式には、漢字の音読み、訓読みだけでなく、各種記号、ギリシア文字、ロシア文字、外字全角文字を表示させるために、次の4種類のカナ文字を入力することができます。

(1) "キゴウ"…………各種記号を羅列します。

(2) "ギリシア"………各種ギリシア文字を羅列します。

(3) "ロシア"…………各種ロシア文字を羅列します。

(4) "ガイジ"…………外字全角文字を羅列します。

変換方法は簡単です。すなわち、変換フィールド内に上記カナ文字列を入力した後 XFER キーを押します。すると、変換フィールドエリアに各種全角文字が羅列されますので、**入門編「5章漢字を表示する」**の項で述べたように、カーソルコントロールキーや数字キーによって目的の文字列を取り出すことができます。

※ 音訓辞書の学習機能について

カナ－漢字変換の音訓変換方式では、使用した漢字を同一グループ内の先頭に配置変換する機能をもっています。したがって、直前に使用した漢字を先頭に羅列します。

この機能は、数字キーによる漢字の選択を行なった場合にのみ行なわれ、カーソルコントロールキーによる選択時には行なわれません。また、同梱の音訓辞書ディスク（マスターディスク）が書き込み禁止状態にある時には働きません。

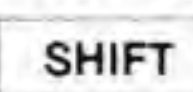
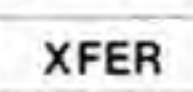
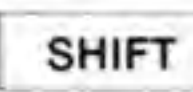
※ 辞書機能を利用する前に...

音訓辞書機能やシステム辞書機能、ユーザー辞書機能を利用して、漢字変換を行なうためには、それぞれ音訓辞書、システム辞書ディスク(オプション：ＣＺ－１１１ＳＦ)、ユーザー辞書ディスク(オプション：ＣＺ－１１１ＳＦ)が必要です。これらの辞書機能を利用する場合は、辞書ディスクを挿入したデバイス名をコンピュータ側に教えなければなりません。そのためには、次の操作を行ないます。

① ＤＥＶＩＣＥコマンドを使って、辞書ディスクを挿入したデバイス名をデフォルトデバイス(注)として指定します。

- フロッピーディスク…０：～３：
- グラフィックメモリ…ＭＥＭ０：，ＭＥＭ１：

② 日本語入力モードにして、漢字変換方式を選択します。選択の仕方は次の通りです。

- 音訓辞書機能を利用する場合　　：SHIFT ＋ F3 キー入力
- システム辞書機能を利用する場合：SHIFT ＋ F4 キー入力
- ユーザー辞書機能を利用する場合：SHIFT ＋ F5 キー入力

このキーを入力することによって、コンピュータ側はそれぞれ辞書機能を利用する際のアクセスを知ります。

(注) デフォルトデバイス…デバイス名の指定を省略したときに用いられるデバイス

以上の操作によって、コンピュータはそれ以後の辞書ディスクをアクセスする際には、指定したデバイスに対してアクセスを行ないます。一度この操作を行なうと、再度同様の操作によって指定を変更しない限り、アクセス先は変わりません。したがって、上記②のキー入力を行なうときは、デフォルトデバイスに何が指定されているのか注意する必要があります。また、各辞書機能を並行して使用する場合は、利用する辞書を変更するたびに上記①、②の操作をくり返してください。

〔例〕音訓辞書ディスクをドライブ1に挿入して利用する場合の操作方法

①DEVICE"1:" ⏎ …… デフォルト・デバイスの指定

② SHIFT + XFER …………………… 日本語入力モードに切換え。詳しくは、本章「2 全角文字の入力方式」を参照してください。

③ SHIFT + F3 …………………… 音訓変換方式の選択

※デバイスを変更しても、音訓辞書の読み出されるデバイスは、変化しません。

以上の操作を応用すれば、音訓辞書の高速利用が可能になります。これは、音訓辞書をそっくりそのままグラフィック用V－RAMに転送することによって行ないます。より高速な音訓変換を行ないたい場合などに用いられます。アクセス速度は、フロッピーディスクよりもグラフィック用V－RAMの方が高速ですので、グラフィック用V－RAM上に音訓辞書を移せば、音訓変換を一瞬のうちに行なうことができます。そのためには、次の操作を行ないます。

① グラフィック用V－RAMの使用目的を、外部記憶用に設定します。

OPTION SCREEN 2 ⏎

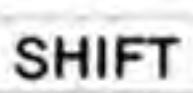
"OPTION SCREEN 3"または"OPTION SCREEN 4"でもかまいません。(詳しくは、『BASICリファレンス・マニュアル』の「OPTION SCREEN」の項参照。)

② グラフィック用V－RAMを初期化します。

INIT"MEM0:" ⏎

とすると、次のメッセージが表示されますので Y キーを押します。

Are you sure ? (y or n)

("OPTION SCREEN 3"に設定した時は、"MEM0:"のかわりに"MEM1:"を実行してください(以後、同じようにします。)。)

③ 音訓辞書（ドライブ0）をグラフィック用V－RAMに転送します。

COPY"0:音訓␣␣変換.DIC"AS"MEM0:" ⏎

④ デフォルト・デバイスをグラフィック用V－RAMに指定します。

DEVICE"MEM0:" ⏎

⑤ 日本語入力モードに切換えます。

SHIFT + XFER

⑥ 音訓変換方式を選択します。

SHIFT + F3

⑦ デバイスの指定を④以前にもどします。

以上で、以後の音訓変換はグラフィック用V－RAMに対してアクセスされますので、非常に高速な漢字変換が行なえます。

注）システム辞書及びユーザー辞書については、グラフィック用V－RAMを利用して高速変換を行なうことはできません。

操作キー一覧

●日本語入力モードにおける特殊キーの動き

キー入力	動　　　　作
SHIFT + XFER または CTRL + XFER	日本語入力モードに切り換えたり、 日本語入力モードから抜け出したりする。
XFER	変換を行なう。
HELP	入力モードの確認。
F1 ～ F10 または GRAPH + 1ヌ ～ 0ワ	入力モードの選択。

●変換フィールドエリア内に入力された状態での働き

キー入力	動　　　　作
⏎	変換フィールドエリア内の文字をテキストエリアに移す。
INS DEL	変換フィールドエリア内の文字を抹消する。
CLR HOME	変換フィールドエリア内の文字をクリアする。
カーソルコントロールキー	無効。
HTAB	連続して漢字を変換する。

●漢字グループを表示している状態

キー入力	動　　　　作
[illegible] [illegible]	漢字グループ内のカーソルを移動する。
[illegible] [illegible]	漢字グループの次候補、前候補を選択する。
ESC	漢字グループの表示を消し、変換フィールド内にカーソルを戻す。
⏎	目的の漢字をテキストエリアに表示する。
テンキー 1 ～ 9	目的の漢字に対応したテンキーを押せばその漢字をテキストエリアに表示する。

6章
フリーエリアについて

本機では、ＣＰＵにＺ８０Ａを採用しています。したがって、メインメモリは６４Ｋバイトのメモリ空間を持っています。このうち、ＢＡＳＩＣインタープリタに使用している領域を除いた残りの部分がフリーエリアになります。

このフリーエリアの領域にプログラムやデータが格納されるわけですが、それらがフリーエリアより大きくなるとメモリが足りなくなり、"Ｏｕｔ ｏｆ ｍｅｍｏｒｙ"エラーになります。

したがって、大きなプログラムを走らせたり、多量のデータを扱うためには、大きなフリーエリアを確保することが必要ですがメインメモリは６４Ｋバイトと制限されていますので、この中でフリーエリアを増やそうとすると、ＢＡＳＩＣを小さくするしかありません。

しかし、ＢＡＳＩＣを小さくすると、その分機能が減り、かえって使いにくいものになってしまいます。

そこで、本機ではグラフィックＶＲＡＭを変数領域として使用できるようにし、その分メインメモリのプログラム領域を増やし、フリーエリアの増大を図っています。

1 フリーエリア

本機のメモリはＢＩＯＳ　ＲＯＭ、メインメモリ、グラフィックＶＲＡＭなどから構成されています。

ＢＡＳＩＣは、このうちのメインメモリにロードされ、残りの領域にプログラムテキストやプログラムに使用される変数が置かれ通常この部分をフリーエリアといいます。

しかし、フリーエリアは、プログラムの実行前と実行後やグラフィックＶＲＡＭを変数に使うか否かなどによって異なります。

そこで、もう少しフリーエリアについて詳しく見ていくことにします。

1.1 ＢＡＳＩＣの起動直後

ＢＡＳＩＣを起動した直後はメインメモリにはＢＡＳＩＣシステムとそれに必要なワークエリアなどがとられるだけです。また、グラフィックＶＲＡＭⅠは変数として使用されるモードとなっていますので、図―１のようになります。

1.2 プログラムロード後（実行前）

プログラムをロードするとＢＡＳＩＣシステムの後ろにプログラムテキストが置かれます。したがって、フリーエリアはメインメモリからＢＡＳＩＣシステムワークエリア、プログラムテキストを除いた部分と、グラフィックＶＲＡＭの部分で図―２のようになります。

1.3 プログラム実行後

プログラムを実行するとプログラムテキストの後ろにプログラムで使用される変数がとられます。またVDIM命令を使っていれば、グラフィックVRAMにも変数がとられます。

この場合のフリーエリアは図―3で示すようにプログラムロードの後のフリーエリアからさらに変数エリアを除いた部分になります。

1.4 グラフィックVRAMを変数として使用しない場合

高解像度グラフィックを使用する場合、グラフィックVRAMのバンク1は、グラフィック描画用に使用され、変数を格納することはできません。また、外部記憶として使用する場合も同様に変数を格納することはできません。

これらの場合、メインメモリの部分のみがフリーエリアとなります。

図―1　BASIC起動直後

メインメモリ

BASIC

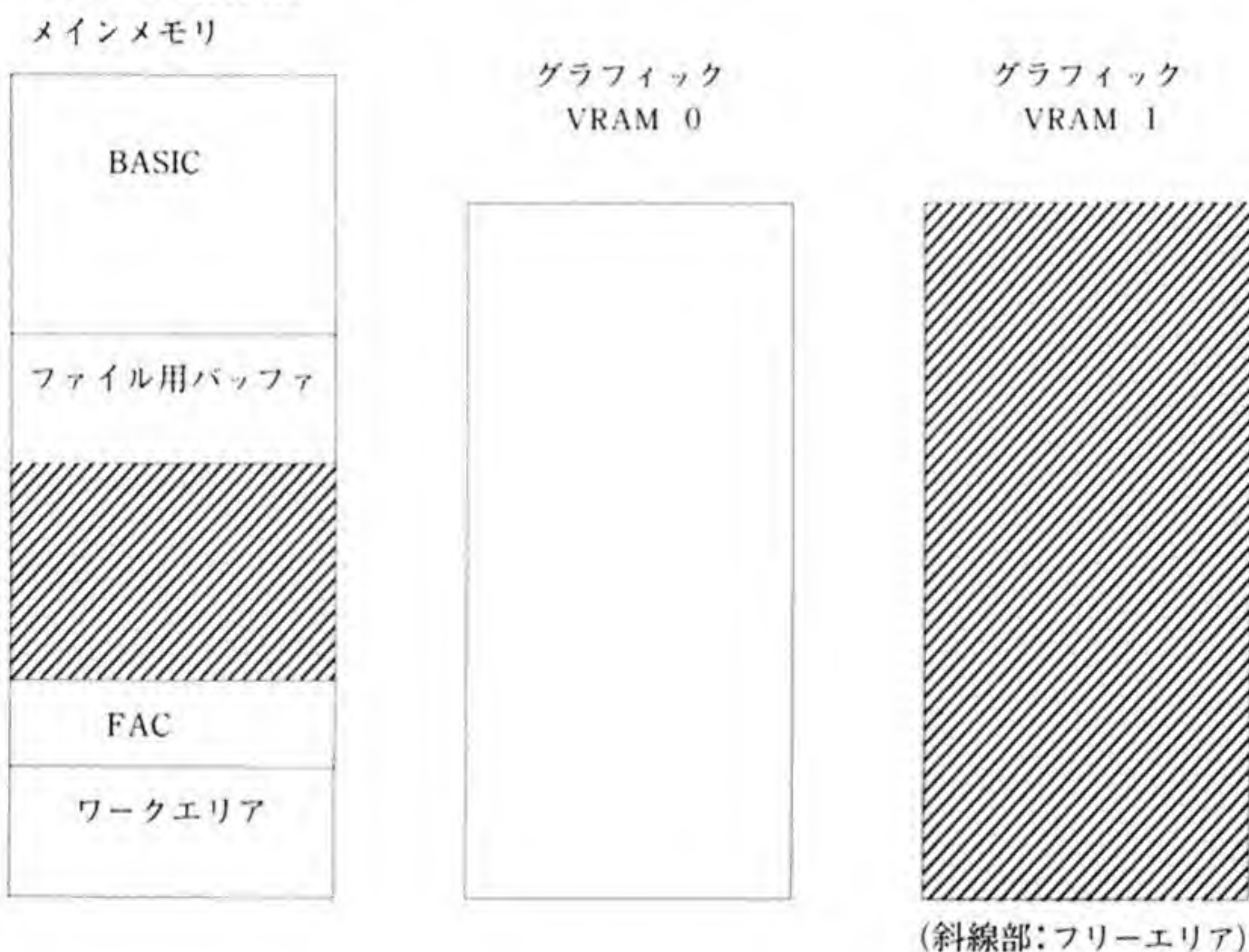
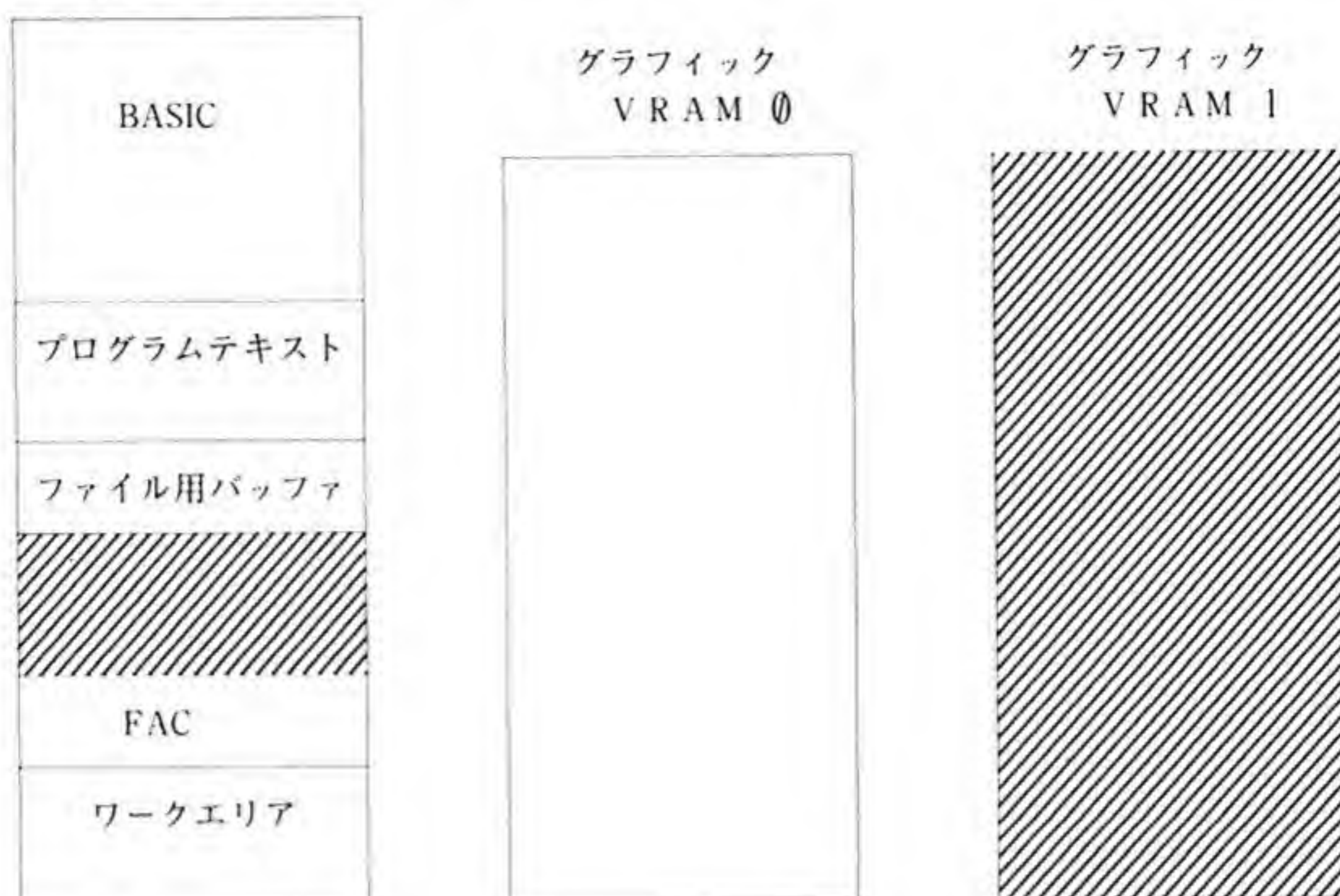
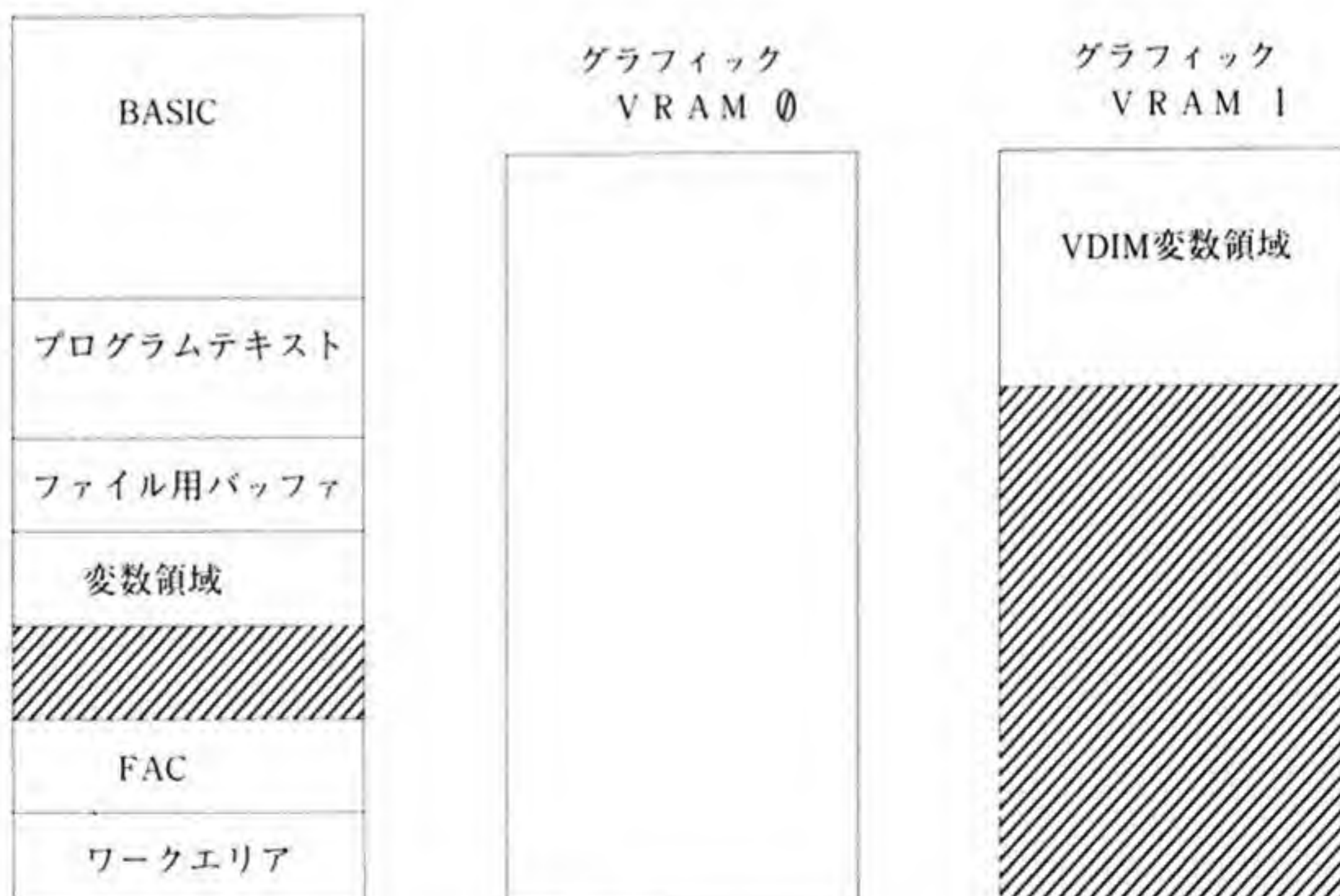
ファイル用バッファ

FAC

ワークエリア

グラフィック
VRAM 0

グラフィック
VRAM 1

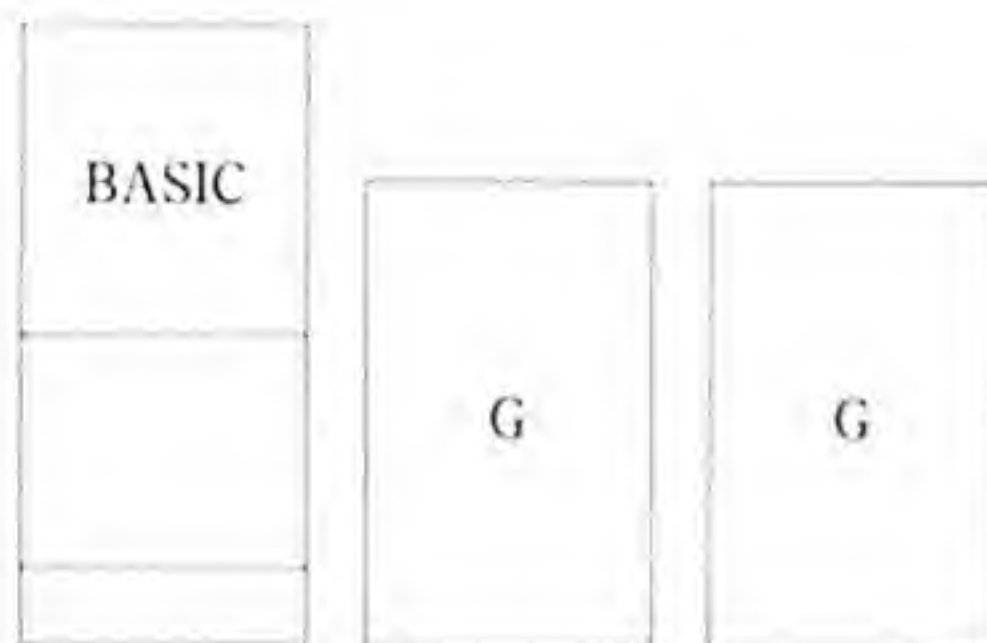
(斜線部：フリーエリア)

図―2　プログラムロード後（実行前）

図―3　プログラム実行後

- ＦＡＣとは演算処理時に使用されるワークエリアです。
- ファイル用バッファはファイルをアクセスする際に使用されるバッファでＭＡＸＦＩＬＥＳ命令により確保される大きさが異なります。

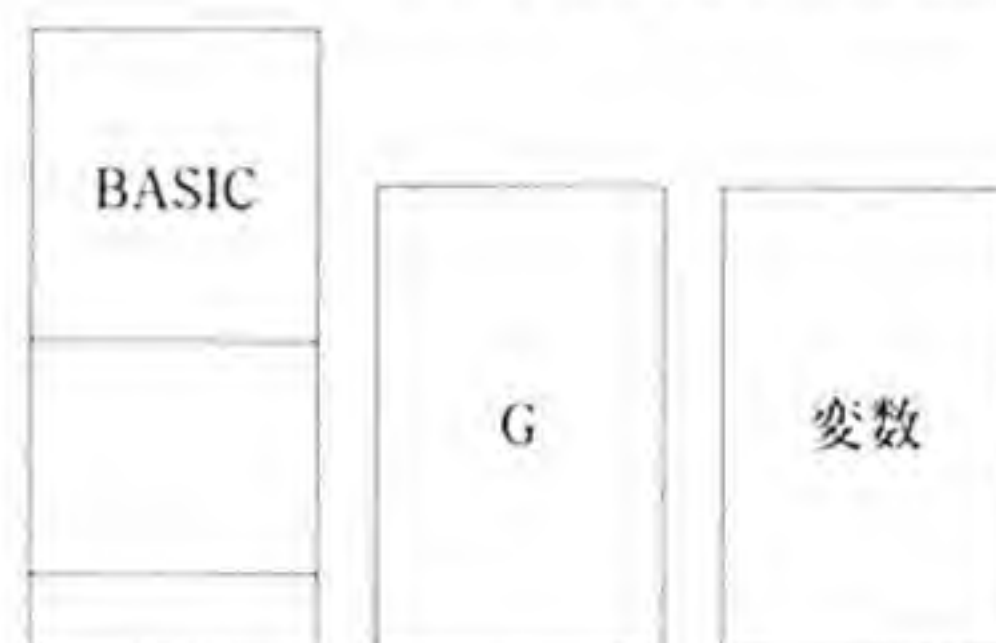
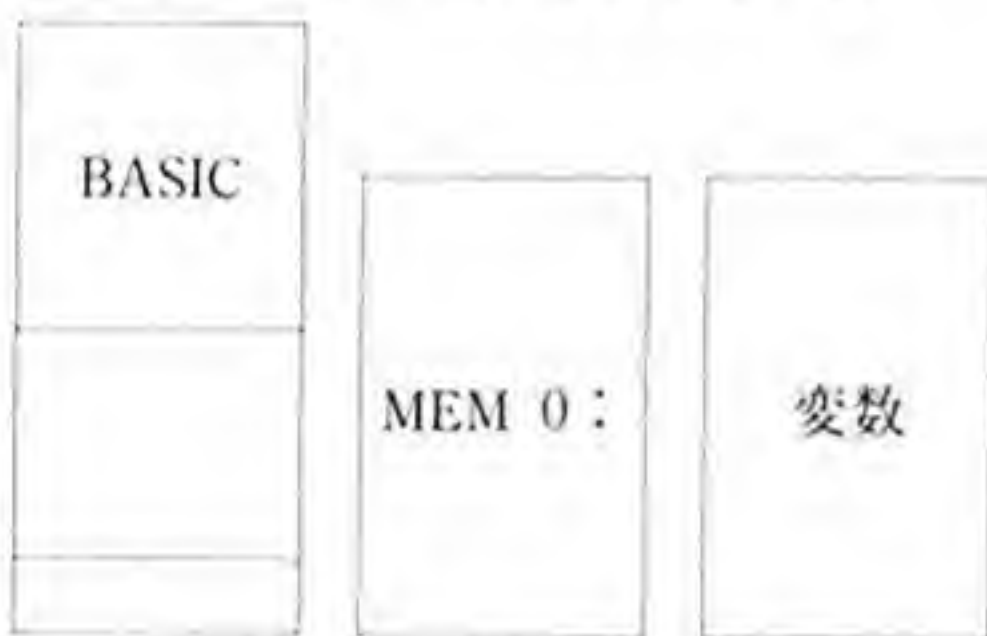
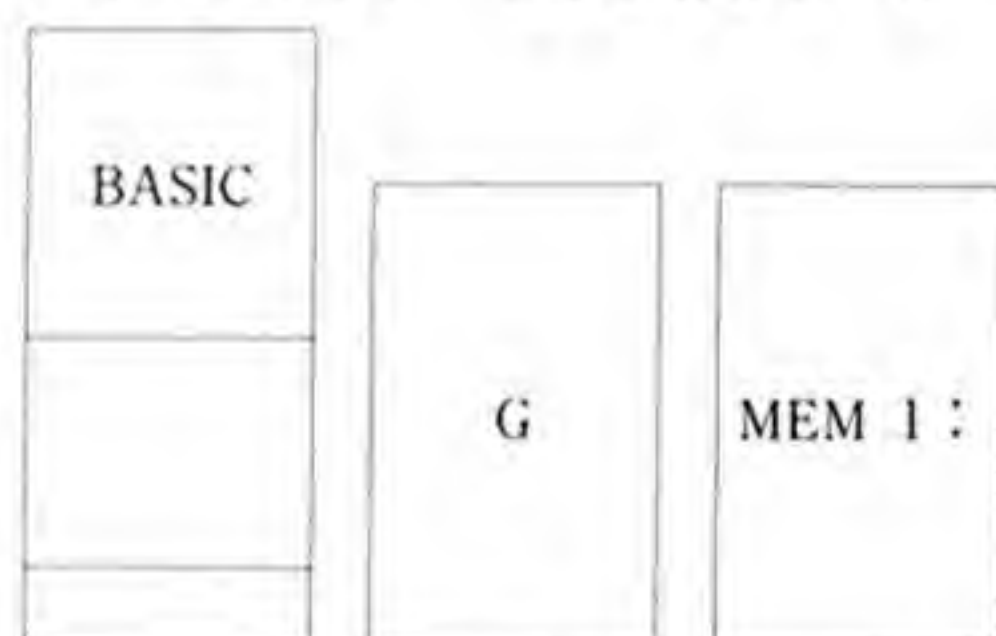
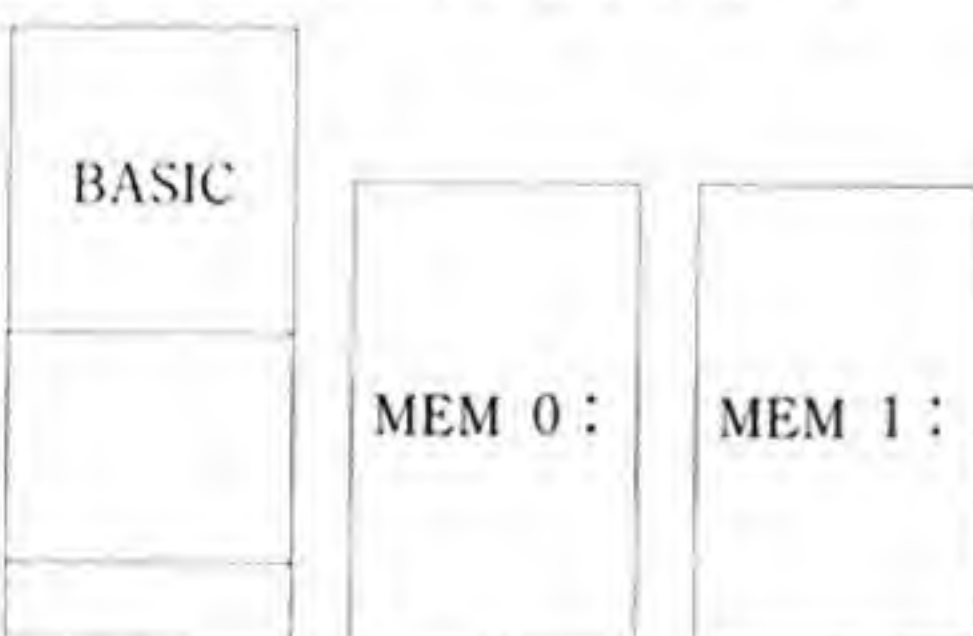
2 オプションスクリーンとフリーエリア

グラフィックＶＲＡＭをどのような用途で使用するかを決めるには**ＯＰＴＩＱＮ　ＳＣＲＥＥＮ**という命令を使います。

ＯＰＴＩＯＮ　ＳＣＲＥＥＮで定義されるスクリーンモードは全部で５つあり、ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　０～ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　４で設定します。

このうち、グラフィックＶＲＡＭを変数領域として使用できるのはＯＰＴＩＯＮ　ＳＣＲＥＥＮ　１とＯＰＴＩＯＮ　ＳＣＲＥＥＮ　２で、それ以外はすべてグラフィックまたは外部記憶とします。

(1)ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　０

BASIC
G
G

(2)ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　１
（ＢＡＳＩＣ起動時の初期状態）

(3)ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　２

(4)ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　３

(5)ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　４

Ｇはグラフィック描画用
ＭＥＭ０：，ＭＥＭ１：はデバイス名を示します。

(1)ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　０

グラフィックＶＲＡＭのバンク０、バンク１ともグラフィックの描画に使用されます。
その結果、グラフィックとしては次の画面モードで使用できます。

６４０×４００	１画面	３２０×４００	２画面
６４０×３８４	１画面	３２０×３８４	２画面
６４０×２００	２画面	３２０×２００	４画面
６４０×１９２	２画面	３２０×１９２	４画面

このモードではVDIM、VDIM　CLEAR、INIT"MEM0："、INIT"MEM1："の命令を実行しようとすると"Bad　screen　mode"エラーとなります。
さらに、640×400、320×400、640×384、320×384の各画面モードにおいては次のステートメントを実行することができません。

　　　OPTION　SCREEN　1
　　　OPTION　SCREEN　2

また、変数、プログラム領域として使用できるのはメインメモリ上のみです。

(2)OPTION　SCREEN　1

グラフィックVRAMのバンク0は、グラフィック描画用として使用し、バンク1を変数領域として使用します。

画面モードとしては、

640×200	1画面	320×200	2画面
640×192	1画面	320×192	2画面

となり、スクリーンの解像度は制限されますが、VDIM命令によりグラフィックエリアに変数を確保することができます。
このモードでのフリーエリアはメインメモリ上の領域とグラフィックVRAMのバンク1を合せた部分となります。

(3)OPTION　SCREEN　2

グラフィックVRAMのバンク0は外部記憶として使用し、バンク1を変数領域として使用します。

したがって、フリーエリアはメインメモリ上の領域とグラフィックVRAMのバンク1を合せた部分となります。
このモードではVDIM、VDIM　CLEAR、INIT"MEM0："は使用できますが、INIT"MEM1："とすると"Bad　screen　mode"のエラーとなります。

(4)OPTION　SCREEN　3

グラフィックVRAMのバンク0はグラフィック描画用として使用し、バンク1を外部記憶として使用するモードです。

プログラムテキスト、変数は共にメインメモリ上にのみ確保されます。

(5)OPTION　SCREEN　4

グラフィックVRAMのバンク0、バンク1共に外部記憶用として用います。

プログラムテキスト、変数は共にメインメモリ上にのみ確保されます。

3　グラフィック描画と外部記憶

OPTION　SCREEN　2、3、4のモードではグラフィックVRAMを外部記憶として使用できますが、このメモリ上にグラフィックデータを描画することも可能です。
すなわち、次の点に注意すれば、グラフィックVRAMを外部記憶として設定した場合でも、グラフィックを描画することもできます。
また、48Kバイトのうち一部を外部記憶として、ファイルを格納し、残りをグラフィック描画にすることも可能となります。

(1)グラフィックVRAMを外部記憶として、設定した場合、WIDTH命令でメモリの内容がクリアされることはありません。

(2)グラフィックＶＲＡＭに対し、ＩＮＩＴ命令を実行すると、グラフィックＶＲＡＭにはメモリ管理のためのデータが書き込まれます。

(3)外部記憶として、使用した場合でも、グラフィック命令によって、グラフィックを描画できますので、ファイルと共存して使用する場合は注意が必要です。

たとえばＣＬＳ命令などによってファイルをこわさないようにしてください。

ファイルはグラフィックＶＲＡＭの上位アドレス（青のメモリ）から順に書き込まれていきますので、上位１６Ｋバイトにファイルを格納し、残り３２Ｋバイト（赤と緑）をグラフィックに使うということも可能です。

4 日本語入力とフリーエリア

ディスクＢＡＳＩＣを起動した時点では、日本語入力は音訓入力モードになっています。音訓入力モードでは音訓辞書ディスクから読み込んで変換を行ないますので、音訓辞書をアクセスするためのルーチンが必要です。

このルーチンはメインメモリ上のＦ０００Ｈ番地からＦ３ＦＦＨ番地を使用するため、

```
CLEAR &HF000
```

となっています。

したがって、音訓変換を使用しない場合には、

```
CLEAR &HF400
```

とすることができますので、フリーエリアが１Ｋバイト増加します。

5 ＮＥＷＯＮとフリーエリア

本機のＢＡＳＩＣはゲーム、ビジネス、教育用などあらゆる用途に対応できるため、ひじょうに多くの命令を含んでいます。

その結果、短いステップで豊富な処理を扱うことができます。

しかし、実際のプログラムではすべての命令を使用しているわけではありません。

たとえば、外部デバイスにディスクしか使わないプログラムではカセットの命令やＲＳ－２３２Ｃの命令は不要です。それらの命令に使用しているメモリ領域をユーザー用のフリーエリアとすればより大きなプログラムやデータが扱えます。

そこで、プログラムで使用しない命令を削除してフリーエリアを確保しようとする場合に使われるのがＮＥＷＯＮ命令です。

この命令によってＢＡＳＩＣをある程度カスタマイズすることができます。

ＮＥＷＯＮで削除される命令群は１０のレベルがあり０～９の番号で指定し番号が小さくなるほどレベルは高くなります。

レベルの高い番号を指定するとそれより低いレベルの命令群はすべて削除されます。

たとえば、ＮＥＷＯＮ０とするとＮＥＷＯＮ０～ＮＥＷＯＮ９のすべての命令が削除されます。

ＮＥＷＯＮと数字（０～９）との間にスペースを入れないでください。

なお、ＢＡＳＩＣ起動時、ＮＥＷＯＮ命令で一度削除された命令はＢＯＯＴ命令もしくはＩＰＬリセットで再度ＢＡＳＩＣを起動しない限り使用できません。

また、一度ＮＥＷＯＮ(ｎ＝０～９)を実行したあと、削除されたＢＡＳＩＣのコマンド、ステートメントを実行しようとしても

```
Reserved feature または Error 34
```

というエラーメッセージが表示されて実行できません。

〈参考〉

① HELP キーを押しながら、ディスクＢＡＳＩＣを起動すると、自動的にＮＥＷＯＮ４が設定されます。

画面１

```
SHARP HuBASIC  CZ-8FB02 Version1.0
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
 80030 Bytes free

Ok
```

②ＢＡＳＩＣを起動後、希望するＮＥＷＯＮを自動的に設定し、すぐにプログラム入力ができるようにするには、"Ｓｔａｒｔ　ｕｐ．Ｂａｓ"のプログラムを次のように変更します。（行番号５７０、６５０を変更）

```
570 IF INKEY$(0)=CHR$(&HE2) THEN 650
```

```
570 IF INKEY$(0)><CHR$(&HE2) THEN 650
```

```
650 COLOR0:KEY0,"NEWON4"+CHR$(13)+"COL.7 ..........
```

↓

０～９の数値を入力するか、または数値を省略して、
希望するＮＥＷＯＮのレベルを設定する。

③変更したプログラムをマスターディスクに

```
SAVE"0:Start up     .Bas"
```

と入力してセーブします。

（例） ＮＥＷＯＮ（省略）に設定すると、ＢＡＳＩＣ起動後は次の画面になります。

画面２

```
SHARP HuBASIC  CZ-8FB02 Version1.0
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
 74493 Bytes free

Ok
```

n	削除されるコマンド、ステートメント
省略	すべてのコマンド、ステートメント、関数など使用可
9	MIRROR$, KANJI$, DTL, RANDOMIZE, WAIT, KEYLIST, KLIST, KBUF, CANVAS, LAYER, TVPW, CHANNEL, VOL,
8	VERIFY, LOAD?, "CAS:", CMT, CMT関数, REW, FAST, EJECT, APSS
7	"COM:", ON COM GOSUB, COM ON/OFF/STOP, POSITION, PATTERN, CIRCLE@, SCROLL, STICK, STRIG, PUSH, POP, ATTR$, RUN"?. Sys"
6	CRT, ON KEY GOSUB, KEY ON/OFF/STOP ON TIME$ GOSUB, TIME$ ON/OFF/STOP
5	MKDIR, CHDIR, RMDIR, HDOFF, SET, NAME, FPOS
4	MOUSE, MOUSE関数, HCOPY
3	GET@, PUT@, CGPAT$, DEVI$, DEVO$, LPOUT CONSOLE#, COPY
2	LIST, LLIST, DELETE, RENUM, AUTO, EDIT, TRON, TROFF, SAVE, SEARCH, KILL, CONT, SAVEM, エラーメッセージ
1	PLAY, PLAY@, MUSIC, MUSIC@, TEMPO, SOUND, SOUND@, CBLACK
0	WINDOW, LINE, PSET, PRESET, CIRCLE, POLY, PAINT, PAINT@, SYMBOL, POINT, PRW, CLICK, NEWONn, PALET, PALET@

6 フリーエリアの大きさを確認するには

フリーエリアはFRE関数によって内容を確認することができます。

FRE関数は

FRE (n)

という書式で書くことができ、nの値が1のときメインメモリのフリーエリア、nの値が2のときグラフィックVRAMのフリーエリア、nの値が0のとき、それらの合計がこの関数の値になります。

7章
ファイルについて

「ファイル」とは、日常、「書類などの情報をバインダーなどにとじたもの」の意味で使われていますが、ＢＡＳＩＣの「ファイル」もこれと同じ意味をもっており、違うのは、情報を記録しておくものが「バインダー」ではなく、「電気や磁気による記憶装置（デバイス）」になっている点です。

ファイルは、一つの単位として取り扱われる関連したレコード（ｒｅｃｏｒｄ）から成り、そこに、プログラムやデータの集合が記録されます。

1 ファイルの種類

ファイルは、アクセス[1]の仕方によって、シーケンシャルアクセスファイルとランダムアクセスファイルに分類されます。

1）アクセス（ａｃｃｅｓｓ）メモリやＩ／Ｏを読み書きすることは、すべてアクセスといいますが、ここでは、「ファイルを読み書きする」の意味で使用しています。

7章

1．1 シーケンシャルアクセスファイル

ファイル中のデータが書き込まれた順に並んでおり、データを読み書きする際には、先頭から順番にしかできないようなファイルのことを、シーケンシャルアクセスファイルといいます。

一般に、このファイルではデータが順序良く並んでいて、無駄がありませんが、特定のデータだけを読み込むことはできません。そのため、データの内容を変更する場合は、内容を変更しないデータも順に読み出し、処理を行なった後、別のファイルに書き込まなければなりません。

1．2 ランダムアクセスファイル

ファイル中の特定のデータの記憶位置を直接指定して読み書きできるようになっているファイルを、ランダムアクセスファイルといいます。

この場合、データは１レコード（＝２５６バイト）単位で読み書きされます。

このファイルでは、任意のデータを指定できるので、シーケンシャルアクセスファイルではできなかった、特定のデータの変更、追加、削除をすることができます。

2 デバイス

バインダーにとじたファイルは書棚にしまって整理するように、コンピュータでも、ファイルを「デバイス」と呼ばれる書棚にしまって整理します。

デバイスには、次のようなものがあります。

- フロッピーディスク
- カセットテープ
- グラフィックメモリ
- 外部メモリ（EMM）
- ハードディスク
- ディスプレイテレビ
- キーボード
- プリンタ
- ＲＳ－２３２Ｃ

このうち、フロッピーディスク、グラフィックメモリ、外部メモリ、ハードディスクなどのデバイスでは、シーケンシャルアクセスファイルおよびランダムアクセスファイルのいずれも使用できますが、カセットテープなどのデバイスでは、シーケンシャルアクセスファイルしか使用できません。（詳しくは「５．デバイス名」の項を参照してください。）

3 ファイルの管理

書棚にしまってあるファイルには見出しを付けておくように、デバイスにしまってあるファイルにも名前が付けられています。

さらに、ファイルを探しやすいように、名前などの情報は、ファイルとは別の場所に保管されています。フロッピーディスクなどのようにランデムアクセスファイルを扱うことのできるデバイスでは、名前などの情報を**ディレクトリ**と呼ばれる場所にまとめて書かれていますが、カセットテープなどのようにシーケンシャルアクセスファイルしか扱うことのできないデバイスでは、１ヵ所にまとめずに各ファイルの先頭に名前などの情報を記録しています。したがって、これから述べようとするディレクトリについては、カセットテープなどのデバイスには利用できませんので、注意してください。

ファイルの管理はこのディレクトリによって行なわれていますが、本機ではそれを**階層ディレクトリ**と呼ばれる方法によって行なっています。

以下階層ディレクトリについて説明します。

現在、パーソナルコンピュータの記憶媒体として、フロッピーディスクが多く使われており、比較的安価で持ち運びに便利なため、保存するファイルの数が増えると、何枚かのフロッピーディスクに分けて記録し、それぞれにインデックスシールを貼って区別するようにします。ところが、ハードディスクのようにフロッピーディスク３０枚分ものデータを扱えるデバイスでは、１台の中にすべてのファイルを保存することができます。このように、何枚ものフロッピーディスクに分けて行なっていたことを１つのハードディスクで行なおうとすると、ファイルの管理上問題が生じてきます。

たとえば、ディスクに入っているファイルのリストを見るためにＦＩＬＥＳコマンドを実行するとします。このとき、ファイルの数が１０や２０であれば、その中から目的のファイルを見つけ出すことはそれほど苦になりませんが、ファイルの数が１００、２００となった場合は、考えただけでも探す意欲がなくなってしまいます。

そこで考え出されたのが階層ディレクトリと呼ばれるファイルの管理方法です。

3.1 ディレクトリ

ディレクトリというのは、ディスクの中に記憶されているファイルの登録簿のことで、ちょうど本の目次に当たります。すなわち、ディスクにおいて、ファイル名やその種類、属性、ディスク上の位置、サイズ、作成年月日などファイルに関する情報が書かれている部分をいいます。

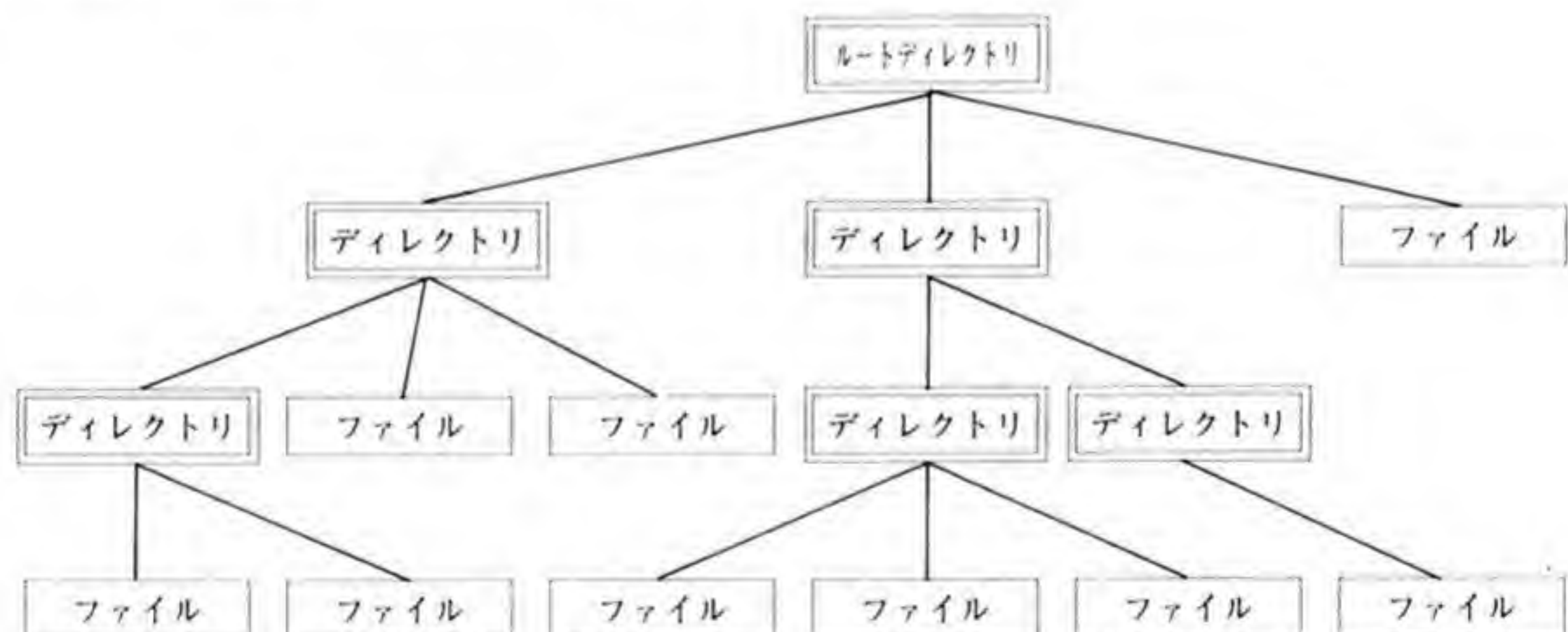
ＦＩＬＥＳコマンドを実行すると、このディレクトリを参照して、その中に含まれているファイル名の一覧表が表示されます。

3.2 単層ディレクトリ

従来のファイルの管理は、ディスク１枚に１つのディレクトリ、すなわち、下図のような単層で行なわれていました。

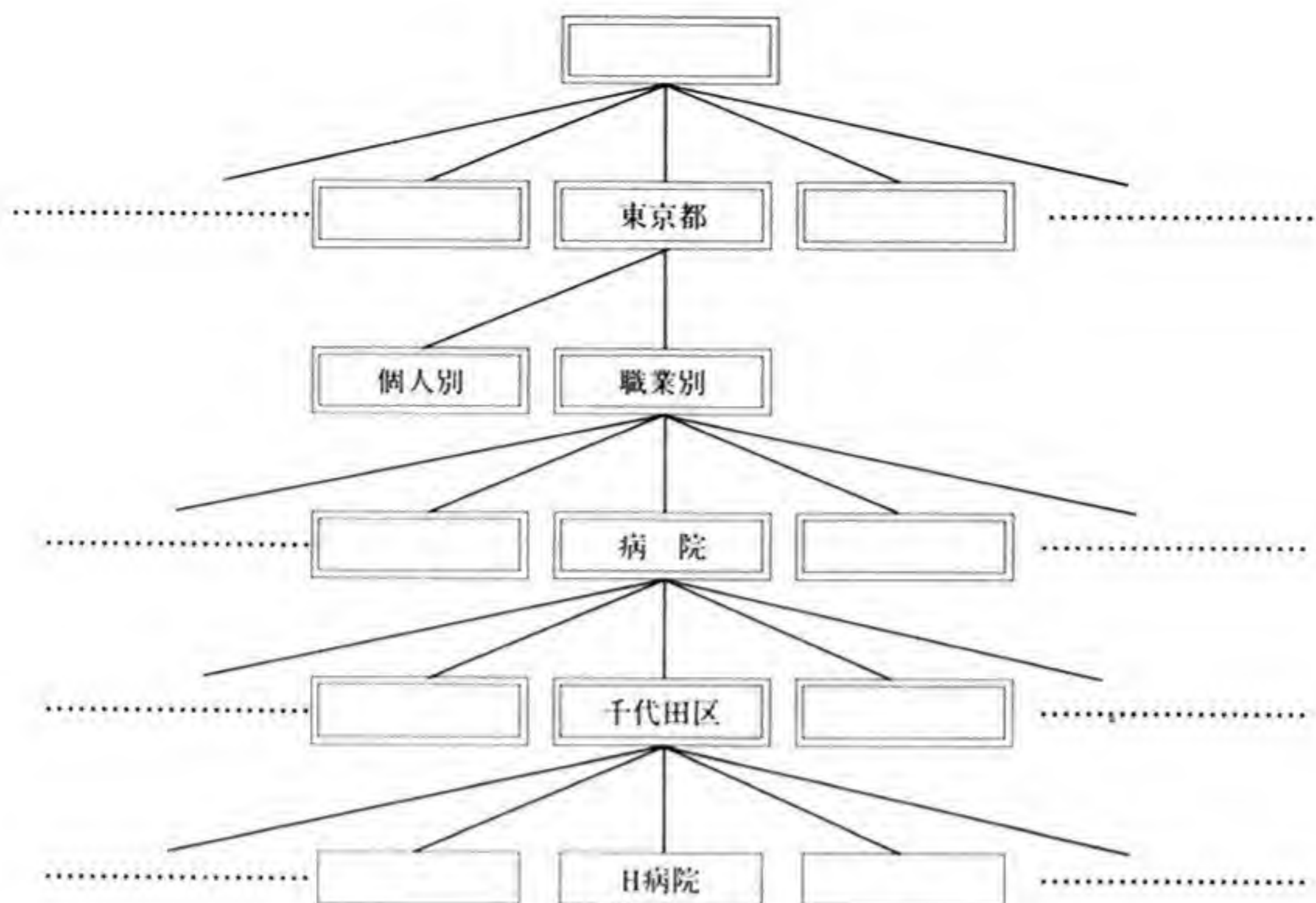
このとき、ＦＩＬＥＳを実行するとファイル１、ファイル２、……、ファイルｎのすべてのファイル名が一度にリストされることになります。

3.3 階層ディレクトリ

容量の大きなデバイス上のファイルを効率的に管理するために考え出されたものが、下図のような木構造をもつ階層ディレクトリです。

たとえば、国内のＨという病院の電話番号を電話帳を使って調べる場合を考えます。Ｈ病院が東京都千代田区にあったとすると、まず東京都２３区の電話帳が必要となります。電話帳は個人別のものと職業別のものとに分かれていますが、病院を調べる場合は職業別のものを使います。次に電話帳の目次から病院の項のページを調べ、そのページを開きます。すると、病院名と住所、電話番号が列記してありますが、各区ごとに項目が分かれています。そこで今度は千代田区の項目を探し出し、さらにその項のところに列記してある病院の中からＨ病院を探し出し、電話番号を知ることができます。つまり、Ｈ病院の電話番号を見つけるまでの経路を示すと、

電話帳の山→東京都→職業別→病院→千代田区→Ｈ病院

となります。

この例を、階層ディレクトリに当てはめると、東京都、職業別、病院、千代田区がディレクトリに対応し、Ｈ病院というのがファイルに対応します。

本機の階層ディレクトリでは階層の経路を表わすのにスラッシュ（／）を使用します。
したがって前記の例は次のように表わされます。

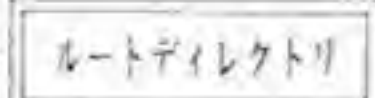
／東京都／職業別／病院／千代田区／

このときファイルにあたるH病院にたどり着くまでの経路をパスといい、ディレクトリを／で区切った

／東京都／職業別／病院／千代田区／

を**パス名**といいます。

また階層ディレクトリの木構造の最上部のディレクトリのことを**ルートディレクトリ**と呼び、先頭の"／"がルートディレクトリを指しています。

3.4 ディレクトリの作成方法

ディスクのフォーマットを行なうとディレクトリが初期化されファイルはすべて消去されます。この状態でＦＩＬＥＳコマンドを実行すると、(ディスクドライブ1ならＦＩＬＥＳ"１："⏎)

```
××  Clusters free
Path name  "×：／"
```

と表示されます。

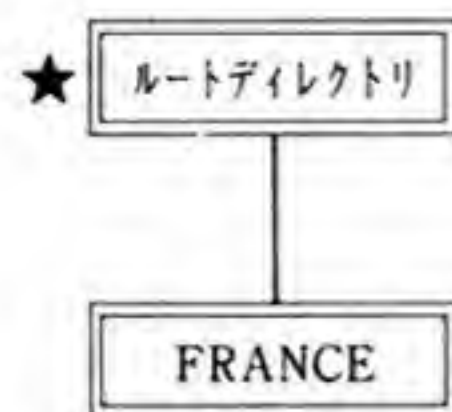
この表示はディスク上にルートディレクトリのみが存在していることを示しています。

★

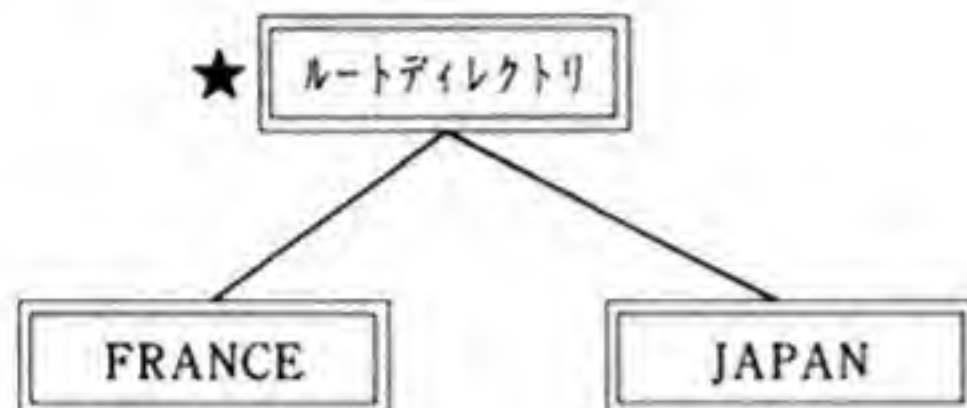
ここに新しくディレクトリを作成する場合、**ＭＫＤＩＲ**というコマンドを使用します。

たとえば、ルートディレクトリの下にＦＲＡＮＣＥというディレクトリを作成する場合、

ＭＫＤＩＲ"ＦＲＡＮＣＥ"⏎

とします。

さらに、ＪＡＰＡＮというディレクトリを追加する場合は、

MKDIR"JAPAN"⏎

を実行します。

もちろん、ここでプログラムファイルやデータファイル（「プログラム、データファイルの保存と再生」参照）を作成することもできます。

たとえば、適当なプログラムを書いてｆｉｌｅ１というファイル名で記録する場合

SAVE"file1"⏎

を実行します。

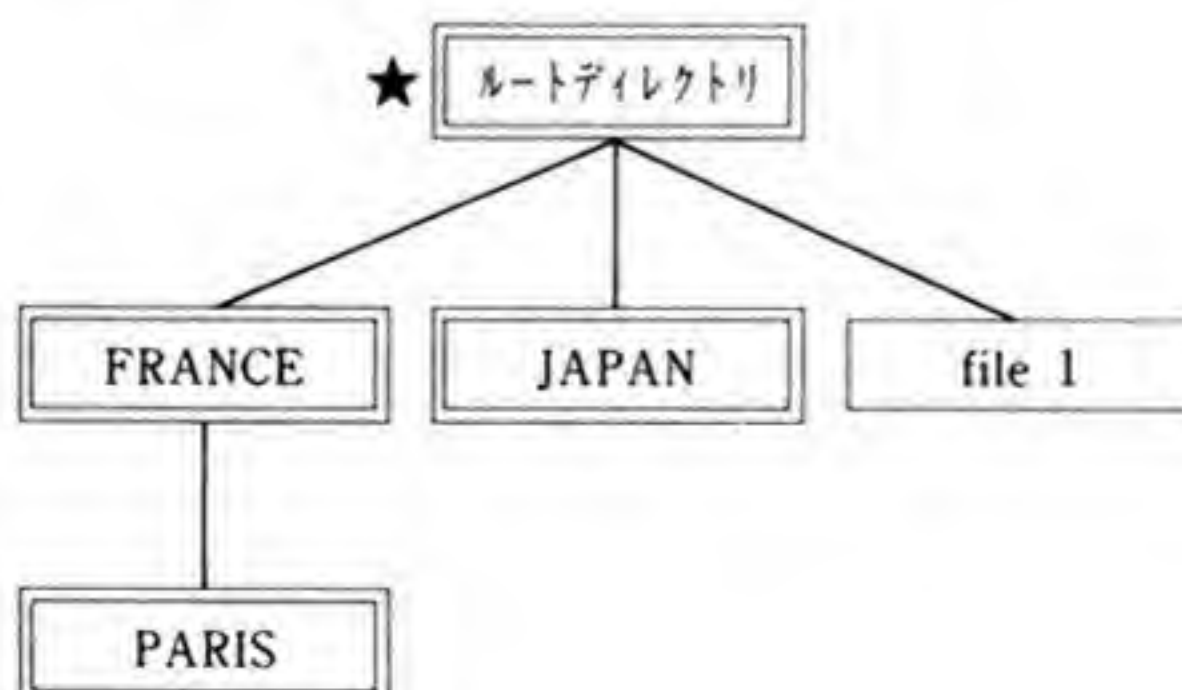
ＦＩＬＥＳコマンドを実行すると、

```
××  Clusters free
Path name    "×:/"
Dir*         "×:FRANCE    .DIR"
Dir*         "×:JAPAN     .DIR"
Bas          "×:file1     .   "
```

と表示されます。

ＦＲＡＮＣＥというディレクトリの下にＰＡＲＩＳというディレクトリを作るには、

MKDIR"FRANCE/PARIS"⏎

を実行します。

さらに下の階層を作る場合も同様に"／"で区切って追加します。

　　MKDIR"……/……/……/……"

3.5 カレントディレクトリの変更

階層ディレクトリ構造では各ディレクトリはそれぞれ独立してファイルのアクセスを行ないます。したがって、同じファイル名のファイルを別々のディレクトリの中に作成することができます。

ファイルをアクセスする場合、そのファイルの存在するディレクトリに移動します。その後はそのディレクトリを起点としてファイルを指定することができます。このディレクトリのことを**カレントディレクトリ**と呼びます。カレントディレクトリの移動には**CHDIR**コマンドを使います。

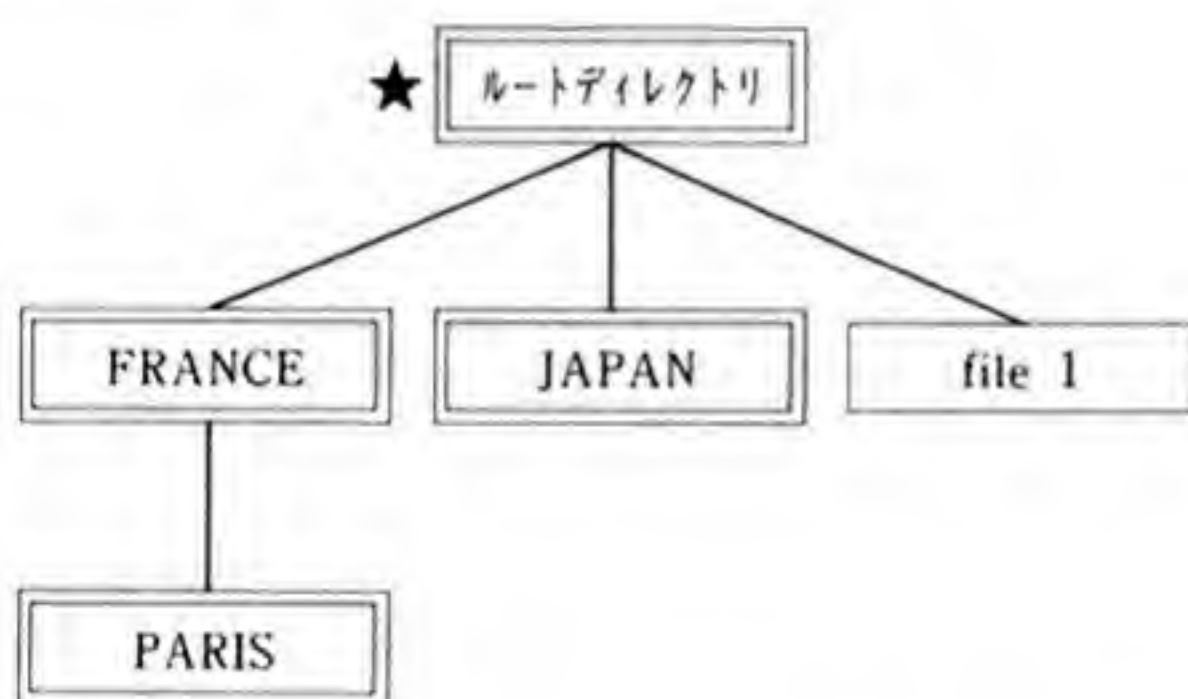
たとえば、カレントディレクトリをルートディレクトリからＦＲＡＮＣＥというディレクトリに移す場合、

　　CHDIR"FRANCE"

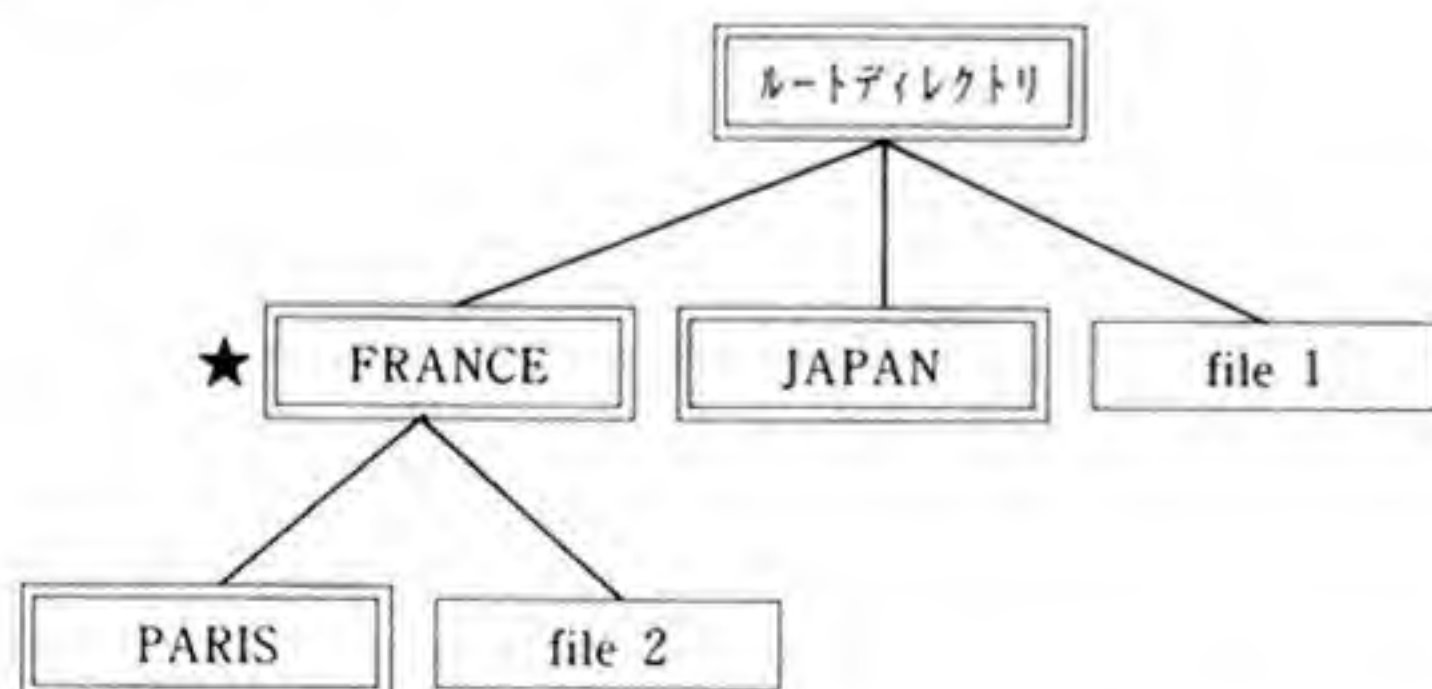
を実行します。

このときＦＩＬＥＳコマンドを実行すると、

```
××  Clusters free
Path name     "×：／FRANCE／"
Dir*          "×：PARIS     .DIR"
```

と表示されます。

ここで、プログラムをfile2というファイル名で記録する場合、

　　SAVE"file2"⏎

とします。

ここからさらにその下のＰＡＲＩＳというディレクトリに移るには、

　　CHDIR"PARIS"⏎

を実行します。

またカレントディレクトリをルートディレクトリに戻すには、

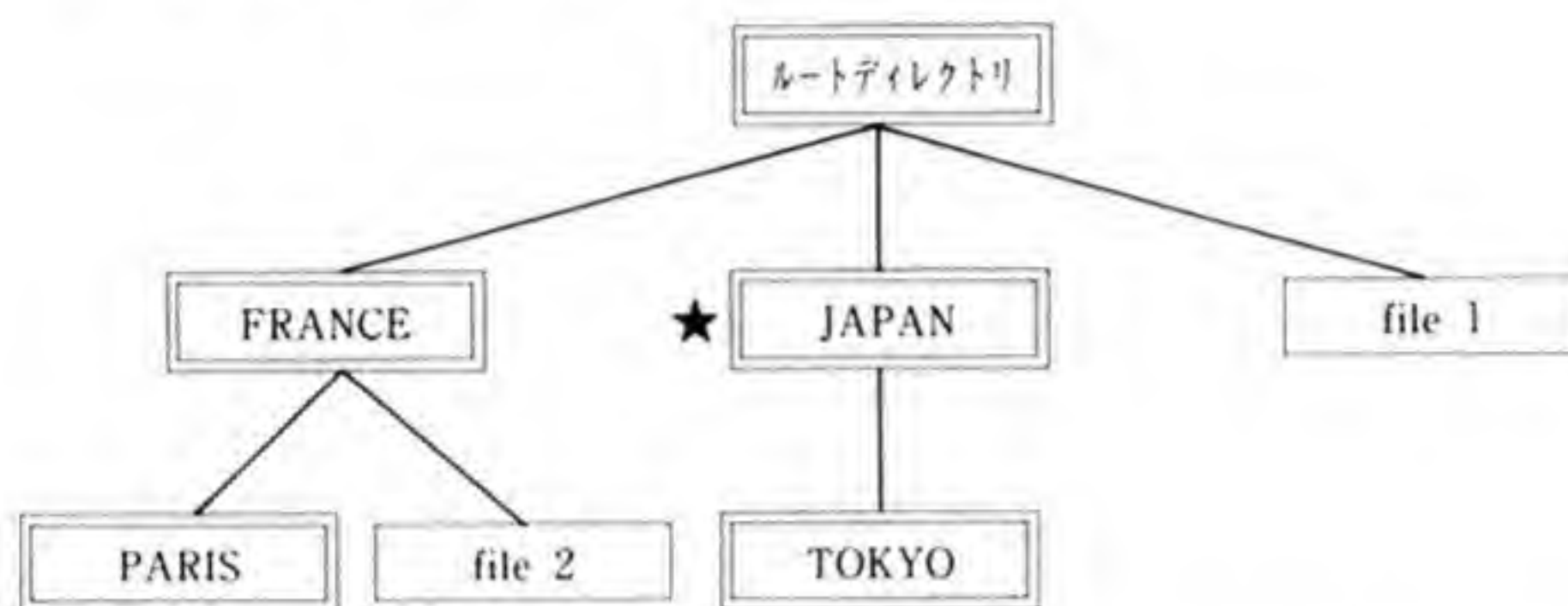
CHDIR"／"⏎

とします。

ＪＡＰＡＮというディレクトリの下にＴＯＫＹＯというディレクトリを作るには、さきほどの方法で、

MKDIR"JAPAN/TOKYO"

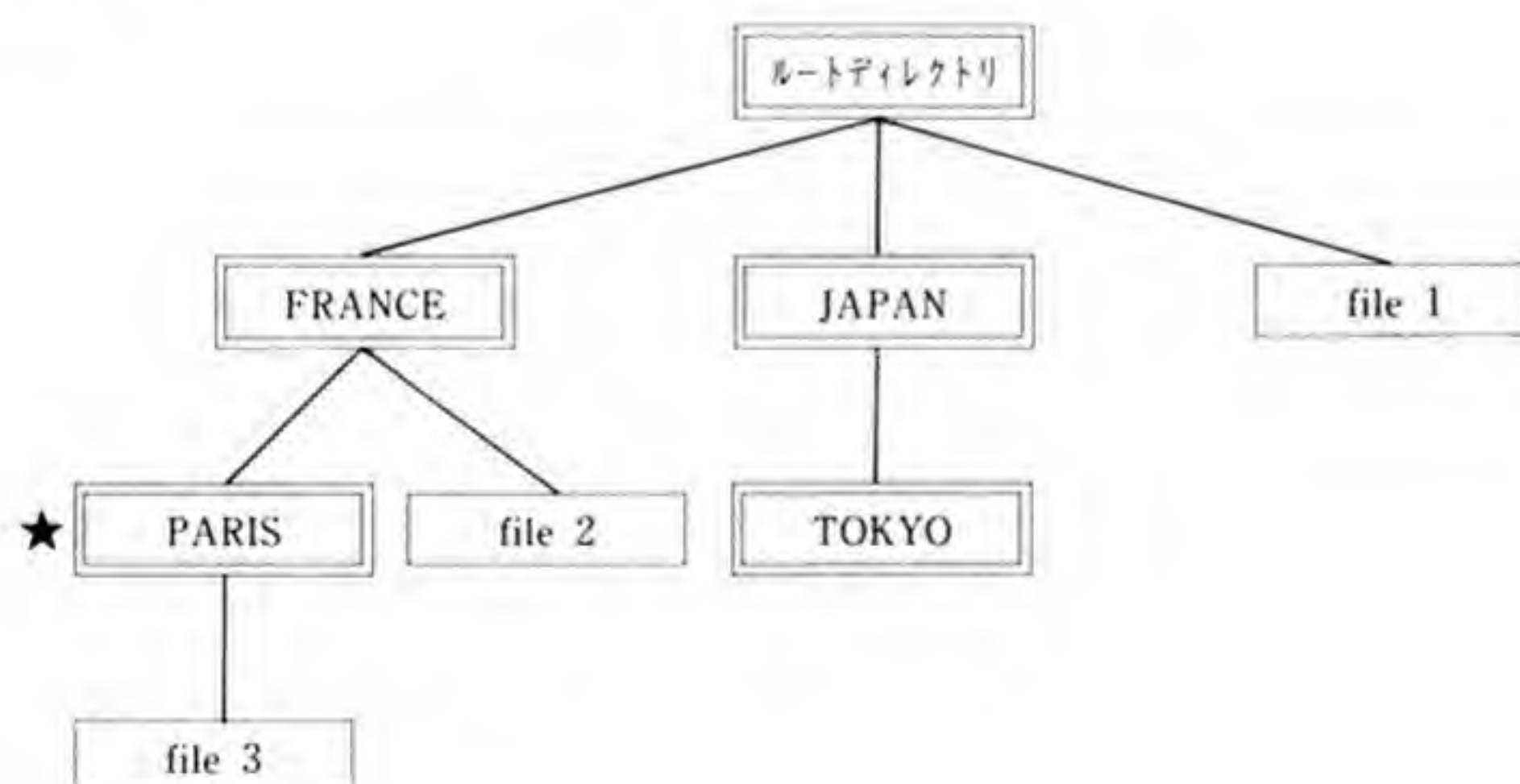
としてもよいのですが、いったんカレントディレクトリをＪＡＰＡＮに移してから作る方法もあります。

CHDIR"JAPAN"⏎
MKDIR"TOKYO"⏎

この状態でＰＡＲＩＳというディレクトリの下にファイルを作りたいときは、

CHDIR"／FRANCE／PARIS"⏎

を実行して、カレントディレクトリをＰＡＲＩＳに移してから、プログラムを作り、

SAVE"file3"⏎

とします。

ここでＦＩＬＥＳを実行すると、

```
××  Clusters free
Path name  "×:／FRANCE／PARIS"
Bas        "×:file3              "
```

と表示されます。

3.6 ディィレクトリの削除

不要なディレクトリを削除するには**RMDIR**というコマンドを実行します。

今、カレントディレクトリをJAPANとして、その下のTOKYOというディレクトリを削除する場合、

```
CHDIR"JAPAN"⏎
RMDIR"TOKYO"⏎
```

とします。

このときディレクトリ"TOKYO"にあるファイルおよびディレクトリはあらかじめ消去されている必要があります。

もしファイルが消去されていなければKILLコマンドで消去してください。

4 ファイルの指定

BASICからファイルをアクセスするときは、シーケンシャルアクセスファイル、ランダムアクセスファイルのいずれの場合も、まず「書棚の所まで行なって、使用したいファイルを取り出し、机の上に運んで開く」という操作をしなければなりません。このことを「ファイルをオープンする」といいます。

コンピュータには、デバイスと呼ばれる何種類かの書棚があり、いくつかのファイルを同時に使用できるように、机が何台か用意されています。

ファイルをオープンするには、書棚と使用したいファイルと机を決定する必要があります。書棚、使用したいファイル、および机を決定することを、**ファイルの指定**といいます。

ファイルの指定は、どの書棚かを示すデバイス名、分類を示すパス名、使用したいファイルを示すファイル名、および机を示すファイル番号によって行ないます。

ファイルの指定は、**ファイルディスクリプタ**と呼ばれる指定子によって行ないます。ファイルディスクリプタは、次の書式で書かれます。

" 〔デバイス名:〕 〔パス名〕 〔ファイル名〕 "

デバイス名:……0:~3:、F0:~F3:、HD0:~HD3:、CAS:
MEM0:、MEM1:、EMM0:~EMM9:、CRT:、SCR:
KEY:、LPT:、COM:

パス名……………〔/〕ディレクトリ 〔;パスワード〕/ディレクトリ 〔;パスワード〕/…/

ファイル名………名前 〔.エクステンション〕〔;パスワード〕

(例) LOAD"0:/FRANCE/PARIS/file1"

(0: → デバイス名、/FRANCE/PARIS/ → パス名、file1 → ファイル名)

OPEN"R",#1,"HD0:/営業部;SH023/PROG1.TST;TAKO"

(HD0: → デバイス名、/営業部;SH023/ → パス名、PROG1.TST;TAKO → ファイル名)

注）ディレクトリの指定は、フロッピーディスクなどのように、ランダムアクセスファイルを扱えるデバイスに対してのみ有効です。

5 デバイス名

デバイス名は、ファイルの存在する書棚（デバイス）を指定するもので、半角文字で最大4文字の文字列とそれに続くコロン（：）で表わされます。

次に示すような種類があります。

デバイス名	指定されるデバイス	用途		
		シーケンシャル入力	シーケンシャル出力	ランダム入出力
0： ｜ 3：	ディスク0（3または5インチ版） ｜ ディスク3（3または5インチ版）	○ ｜ ○	○ ｜ ○	○ ｜ ○
F0： ｜ F3：	ディスク0（8インチ版） ｜ ディスク3（8インチ版）	○ ｜ ○	○ ｜ ○	○ ｜ ○
HD0： ｜ HD3：	ハードディスク0 ハードディスク3	○ ｜ ○	○ ｜ ○	○ ｜ ○
CAS：	カセットテープ	○	○	
MEM0： MEM1：	グラフィックメモリ0 グラフィックメモリ1	○ ○	○ ○	○ ○
EMM0： ｜ EMM9：	外部メモリ0 ｜ 外部メモリ9	○ ｜ ○	○ ｜ ○	○ ｜ ○
CRT： SCR：	ディスプレイテレビ ディスプレイテレビ		○ ○	
KEY：	キーボード	○		
LPT：	プリンタ		○	

通信デバイス

COM：	RS-232C　ポート	○	○	

＊"COM："は特にシーケンシャル入出力として使用することができます。

6　パス名

パス名は、アクセスしようとするファイルが木構造をもつディレクトリのうちのどこにあるかを示します。

パス名は、次の規則に従ってつけられます。

a）パス名はスラッシュ（／）によっていくつかの階層部分に区切られています。

／ディレクトリ／ディレクトリ／………／

b）1つのディレクトリは、半角文字で1～13文字の文字列で表わされています。このとき、全角文字は、1文字で2文字分と数えます。

c）各ディレクトリの後ろにセミコロン（；）をつけて、続けてパスワードを指定することができます。パスワードはファイル内容の秘密を守るためのもので、いったん指定すると、以後パスワードを省略してファイルをアクセスすることができなくなります。

／ディレクトリ；パスワード／………／

d）デバイス名、ディレクトリ、セミコロン（；）、パスワードを合わせて、半角文字128文字以内でなければなりません。

デバイス名：／ディレクトリ；パスワード／………／

128文字以内

e）パス名の先頭のスラッシュ（／）はルートディレクトリを指しています。したがって、先頭のスラッシュを省略すると、現在のカレントディレクトリの下のディレクトリを指定することになります。

f）パス名の指定の仕方が違うときは、「Ｂａｄ　ｆｉｌｅ　ｄｅｓｃｒｉｐｔｅｒ」のエラーが出ます。

（例）／営業／鈴木／

／営業／鈴木；ＡＹＡＫＯ／

／技術；Ｔ００７／田中；Ｙｏｓｈｉｏ／

注）パス名の指定はフロッピーディスクなどのように、ランダムアクセスファイルを扱えるデバイスに対してのみ有効です。

7　ファイル名

ファイル名は、使用するファイルの名前で、デバイス名で指定されたデバイス上のどのファイルかを示します。

ファイル名は、次の規則に従ってつけられます。

a）ファイル名は3つの部分に区切ることができます。

名　　前〔．エクステンション〕〔；パスワード〕

b）名前は、半角文字で1～13文字の文字列、エクステンションは1～3文字で表わされます。ファイル名は、名前、ピリオド、エクステンション、セミコロン、パスワードをあわせて31文字以内でなければなりません。

このとき、全角文字は1文字で2文字分と数えます。

名前．エクステンション；パスワード

31文字以内

c）名前が１３文字より長かったり、エクステンションが3文字より長い場合およびファイル名が３１文字を越えた場合は、「Ｂａｄ ｆｉｌｅ ｄｅｓｃｒｅｐｔｅｒ」のエラーが出ます。

d）パスワードはファイルの機密を保持するためにつけ、いったん指定すると以後パスワードを省略してファイルをアクセスすることができなくなります。

（例）ＴＥＳＴ．Ａｓｃ
在庫管理．Ｂｉｎ
Ｔｅｌｅｐｈｏｎｅｌｉｓｔ．Ｂａｓ；ＫＡＭＩＳＡＮ

8 ファイル番号

ファイル番号は、ファイルのオープン時、用意される机の番号で、０～１５の整数で表わされます。使用する机（ファイル）の数は、ＭＡＸＦＩＬＥＳステートメントで指定することができます。
（例）ＭＡＸＦＩＬＥＳ　２……１、２の２つの机（ファイル）を使用することができます。

8章
プログラム、データファイルの保存と再生

本機のメインメモリにキーボードから入力したプログラムは、いったん電源を切ると、メインメモリ上には保存されません。そこで、カセットテープやフロッピーディスクなど外部ファイルへの保存が必要になります。また、プログラム中で多くのデータを処理したい場合にも、外部ファイルにデータの保存・再生が必要になります。ＢＡＳＩＣでは、このプログラム、データの保存・再生をそれぞれプログラムファイル、データファイルという形で行ないます。ここではプログラム、データの保存および再生とそれに関するコマンドを説明します。

1 プログラムファイル

コンピュータのメインメモリのプログラムは、カセットテープやフロッピーディスクなどの外部ファイルにプログラムファイルとして保存・再生が可能です。その保存・再生は次のコマンドを使用して行ないます。

ＳＡＶＥ、ＬＩＳＴ、ＳＡＶＥＭ、ＬＯＡＤ、ＬＯＡＤＭ、ＶＥＲＩＦＹ、ＬＯＡＤ？、ＩＮＩＴ、ＯＰＴＩＯＮ ＳＣＲＥＥＮ、ＫＩＬＬ、ＤＥＶＩＣＥ、ＦＩＬＥＳ、ＬＦＩＬＥＳ、ＲＵＮ、ＭＥＲＧＥ、ＣＨＡＩＮ

8章

1.1 プログラムの保存

メインメモリ内のプログラムを外部ファイルに記録することを、プログラムの保存（セーブ）といい、ＳＡＶＥ、ＬＩＳＴ、ＳＡＶＥＭコマンドによって実行できます。

ＳＡＶＥ〔"〔デバイス名：〕ファイル名"〕〔,Ａ〕

デバイス名……０：～３：、ＣＡＳ：、ＭＥＭ０：～ＭＥＭ１：、ＥＭＭ０：～ＥＭＭ９：
Ｆ０：～Ｆ３：、ＨＤ０：～ＨＤ３：、ＣＯＭ：

メインメモリ内のＢＡＳＩＣプログラムを指定されたデバイスへ指定されたファイル名を付けてセーブします。また、最後にＡを指定するとプログラムを**アスキー形式**でセーブします。アスキー形式とはプログラムをキャラクタコード（アスキーコード）に変更してセーブすることです。Ａを指定しないときが一般的な記録方式であり、プログラムは中間コードに変換されてセーブされます。アスキーセーブは一般的なセーブよりも多くのファイル領域が必要になりますが、後でのべるＭＥＲＧＥコマンドで使用することができます。

〔注意〕①ファイル名は、プログラムをセーブするときにファイルにつける名前です。名前は、英数字、カナ、記号、日本語を使用し半角文字で１３文字（２バイト文字は２文字として扱います）までです。なお、名前の中に .（ピリオド）"（引用符）:（コロン）;（セミコロン）/（スラッシュ）は使用できません。

②グラフィックメモリにプログラムをセーブする場合、あらかじめ次の2つのコマンドを実行しておく必要があります。

OPTION SCREEN n （n：2～3）

グラフィックメモリを外部記憶装置として使用することを定義します。その場合、nの値は2～4を指定します。詳しくは、『BASICリファレンスマニュアル』を参照してください。

INIT"MEM{0/1}:"

グラフィックメモリをイニシャライズしセーブできる状態にします。ダイレクト命令で実行した場合、「Are you sure? (y or n)」と表示されますので[Y]キーを入力してください。

③ディスクにプログラムをセーブする場合、ディスクをフォーマットしておく必要があります。ディスクのユーティリティプログラムを参照してください。

④デバイス名にCOM：を指定した場合には、RS―232Cポートへのプログラムの送り出しが行なわれます。この場合には、Aオプションをつけなくてもアスキー形式でプログラムを送り出します。

では、実際にプログラムをセーブしてみましょう。次のプログラムを「X1―WORLD」という名前でセーブしてみることにします。

〔例〕

```
10 INPUT "N=";N
20 FOR P=1 TO N
30 PRINT "X1-WORLD"
40 NEXT P
50 END
```

＊プログラムをRUNすると、回数をきいてきますから、数字を入力してください。入力した回数だけ「X1―WORLD」と表示します。

上のプログラムを入力後、次のようにしてセーブを行ないます。

•カセットテープにプログラムをセーブする場合

```
SAVE"CAS:X1—WORLD"⏎
Ok
```

•フロッピーディスク（ドライブナンバー1）にプログラムをセーブする場合

```
SAVE"1:X1—WORLD"⏎
Ok
```

•そのセーブをアスキー形式で行なう場合

```
SAVE"1:X1—WORLD",A⏎
Ok
```

•グラフィックメモリにセーブする場合

```
OPTION SCREEN2⏎
Ok
INIT"MEM0:"⏎
Are you sure? (y or n) Y⏎
Ok
SAVE"MEM0:X1—WORLD"⏎
Ok
```

LIST〔デバイス名：ファイル名"〔,〕〕〔〔開始行番号〕〔ー〔終了行番号〕〕

デバイス名…ＳＡＶＥコマンド指定できるもの以外に"ＳＣＲ："、"ＣＲＴ："（どちらも画面)、"ＬＰＴ:"（プリンタ）があります。

このコマンドを実行することによりメインメモリ内のプログラムを指定されたデバイスへ指定されたファイル名を付けてアスキー形式でセーブすることができます。「ＳＡＶＥ"ファイルディスクリプタ"，Ａ」と異なる点はセーブする行番号を指定できることです。すなわち、メインメモリにあるプログラムの開始行番号から終了行番号まで（開始行番号だけを指定した場合はその行番号だけ）をセーブすることができます。このコマンドと後でのべるＭＥＲＧＥコマンドを合わせてもちいることによって、あるプログラムのサブルーチン（プログラムの一部でまとまった作業を行なう部分）を他のプログラムに追加することができます。

また、すべてのパラメータを省略することによってメインメモリ中のプログラムリストを画面に表示することができます。

〔例〕 **LIST"CAS:EXP", 30-50**⏎
Ｏｋ

この実行により、メインメモリにあるプログラムの行番号３０～５０までが、アスキー形式でカセットテープにセーブされます。

〔例〕 **LIST"0:TEST", 10-40**⏎
Ｏｋ

この実行によりメインメモリにあるプログラムの行番号１０から４０までがアスキー形式でフロッピーディスクのドライブ０にセーブされます。

SAVEM〔"〔デバイス名：〕ファイル名"〕，開始アドレス，終了アドレス〔，実行開始アドレス〕

＊デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

メインメモリ内の機械語プログラムを外部ファイルにセーブします。

〔注意〕 開始アドレス、終了アドレスは機械語プログラムをセーブするときにその範囲を指定するもので、メインメモリ上の番地を指定します。これはファイルにも記録され、再生のときに使われます。

実行開始アドレスは機械語プログラムを実行するときの開始場所を指定するもので、省略するとセーブの開始アドレスが実行開始アドレスになります。

〔例〕 **SAVEM"CAS:TEST", &H8000, &H80FF, &H800E**

メインメモリ内の＆Ｈ８０００から＆Ｈ８０ＦＦにある実行番地＆Ｈ８００Ｅのプログラムをカセットにセーブします。

1.2 プログラムの再生

カセットテープ、フロッピーディスクなどの外部ファイルにセーブされたプログラムをメインメモリに移すことをプログラムの再生（ロード）といいます。ロードはＬＯＡＤ、ＲＵＮ、ＬＯＡＤＭコマンドによって実行できます。

LOAD〔"〔デバイス名：〕ファイル名"〕

デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

外部ファイルに記録されているＢＡＳＩＣプログラムをメインメモリにロードします。

〔注意〕①プログラムをロードすると、それまでメインメモリにあったプログラムは消えます。またファイル名はセーブしたときと、全く同じでなければなりません。もし、指定したファイル名のファイルがない場合には「File not found」のエラーメッセージを表示します。

②デバイス名にＣＯＭ：を指定した場合にはＲＳ－２３２Ｃポートよりプログラムを読み込みます。この場合、通信回線より送られてくるプログラムはアスキー形式でなければなりません。

〔例〕ＳＡＶＥコマンドの所でセーブしたファイル名"Ｘ１－ＷＯＲＬＤ"のプログラムをロードします。

カセットテープからプログラムをロードする場合、まずプログラムの先頭まで巻戻し（または早送り）した後、次のコマンドを入力します。

LOAD"CAS：X1－WORLD"↵

Ｏｋ

フロッピーディスク（ドライブナンバー１）からプログラムをロードする場合、

LOAD"1：X1－WORLD"↵

Ｏｋ

＊なおアスキー形式でセーブされたプログラムも同様にしてロードできます。

グラフィックメモリからプログラムをロードする場合、

LOAD"MEM0：X1－WORLD"↵

Ｏｋ

LOADM〔"〔デバイス名：〕ファイル名"〔，ロード開始アドレス〕〕 LOADM〔"〔デバイス名：〕ファイル名"〔，〔ロード開始アドレス〕，R〕

デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

外部ファイルに記録されている機械語プログラムをメインメモリにロードします。最後のパラメータＲを指定した場合ロード終了後プログラムを直ちに実行します。

〔注意〕ロード開始アドレスを指定すると、そのアドレスから機械語プログラムのロードを開始し、この開始アドレスを省略した場合、ＳＡＶＥＭで指定したセーブ開始アドレスからロードが始まります。

〔例〕 LOADM"CAS：TEST"↵

Ｏｋ

カセットにセーブされた"ＴＥＳＴ"という機械語プログラムがロードされます。

LOADM"CAS：TEST"，，R↵

Ｏｋ

"ＴＥＳＴ"という機械語プログラムがロードされた後、直ちに実行されます。

```
RUN〔"〔デバイス名:〕ファイル名"〕
```

＊デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

外部ファイルに記録されているＢＡＳＩＣプログラムをメインメモリにロードした後、直ちに実行します。また、すべてのパラメータを省略した場合、メインメモリ中のプログラムを実行します。

〔例〕 **RUN"1:X1−WORLD"**⏎

フロッピーディスク（ドライブナンバー１）の"Ｘ１−ＷＯＲＬＤ"プログラムをロードした後、直ちに実行します。

1.3 プログラムのベリファイ

カセットテープにセーブしたプログラムをメインメモリにあるプログラムとくらべて、正しくセーブされたかどうかを調べることをプログラムのベリファイといいます。このときに使うのが、ＶＥＲＩＦＹまたはＬＯＡＤ？です。この２つは全く同じ意味のコマンドです。

```
VERIFY〔"〔CAS:〕ファイル名"〕
LOAD?  〔"〔CAS:〕ファイル名"〕
```

＊このコマンドで扱えるデバイスは"ＣＡＳ："のみです。

カセットテープにセーブしたプログラムが正しく記録されているかどうかを調べます。なお、このコマンドの実行はカセットテープにプログラムをセーブ後すぐに行なってください。

〔例〕ここではＳＡＶＥコマンドの項で示したプログラム「Ｘ１−ＷＯＲＬＤ」をカセットテープにセーブした後、ベリファイします。

①プログラムを入力します。

②カセットテープに"Ｘ１−ＷＯＲＬＤ"をセーブします。

プログラムをセーブできるカセットテープをデータレコードにセットします。テープカウンタを０にしておくか、カウンタ番号を紙に書いておけばカセットテープのどこにセーブしたかの目印になります。

SAVE"CAS:X1−WORLD"⏎

と入力してください。画面にＯｋが表示され、カーソルが点滅したらセーブ終了です。

③テープをセーブした位置まで巻きもどし、次のように入力します。

VERIFY"CAS:X1−WORLD"⏎

正しくセーブされていれば次のように表示されます。

```
Found"X1−WORLD"
Ok
```

正しくセーブされていなかった場合は次のように表示されます。

```
Tape read error
```

この場合、もう一度セーブしてください。

1.4 ファイルの削除

メインメモリにあるプログラムを消すにはＮＥＷを使いますが、外部ファイルに記録したプログ

ラムを消すには、ＫＩＬＬという命令を使います。

```
KILL"〔デバイス名：〕ファイル名"
```

デバイス名……Ø：～3：、MEMØ：～MEM1：、EMMØ：～EMM9：、FØ：～F3：、HDØ：～HD3：

指定したデバイス内の指定したファイルを削除します。このコマンドによって削除されたプログラムはもとにもどりません。十分注意して行なってください。

〔注意〕デバイス名に"CAS："を指定して、カセットに記録されたプログラムを消すことはできません。

〔例〕 **KILL"1：X1−WORLD"**⏎
Ok

フロッピーディスク(ドライブナンバー1)にセーブされた"X1−WORLD"プログラムを削除します。

1.5 ファイル名の一覧表

外部ファイルにセーブされたファイル名の一覧表はＦＩＬＥＳ、ＬＦＩＬＥＳのコマンドを実行することによって与えられます。

```
FILES〔"デバイス名："〕
```

```
LFILES〔"デバイス名："〕
```

デバイス名……Ø：～3：、CAS：、MEMØ：～MEM1：、EMMØ：～EMM9：、FØ：～F3：、HDØ：～HD3：

デバイス名で指定したデバイス内にあるファイル名の一覧表がＦＩＬＥＳの場合は画面に表示され、ＬＦＩＬＥＳの場合はプリンタに出力されます。

〔参考〕ＦＩＬＥＳを行なった後、その中に含まれるファイルをロード、削除、実行したい場合は次のようにすると簡単に行なえます。
(1)そのファイル名の初めに表示されたＢasまたはＡscの所へ、カーソルを移動する。
(2)ＢasまたはＡscのかわりにＬＯＡＤ、ＫＩＬＬ、ＲＵＮを入力し⏎キーを押す。

〔例〕 **FILES"1："**⏎
フロッピーディスク（ドライブナンバー1）のファイルが出力されます。
＊ファイルリストの初めに表示されるＢas、Ａsc、Ｂin、Ｄirには次の意味があります。
Ｂas……ＢＡＳＩＣプログラムのファイル
Ａsc……アスキー形式で書き込まれたファイル
Ｂin……機械語プログラムのファイル
Ｄir……ディレクトリのファイル

〔注意〕カセットテープなどでは、ディレクトリのファイルを扱うことはできません。
また、カセットテープに対してＦＩＬＥＳおよびＬＦＩＬＥＳを実行するとデバイス名

が"ＣＡＳ0："と表示されますが、"ＣＡＳ："と"ＣＡＳ0："とは全く同じものです。

1.6 デバイス名の省略

ファイルに関するコマンドやステートメントを使うプログラムにおいて、何度も同じデバイス名を指定しなければならないことがあります。そのような時ＤＥＶＩＣＥコマンドを使用してデバイス名を省略した場合の指定が行なえます。

```
DEVICE"デバイス名"
```

デバイス名……すべてのデバイス名が指定できます。

ＣＲＴ：、ＳＣＲ：、ＫＥＹ：、ＬＰＴ：、ＣＡＳ：、0：～３：、
Ｆ0：～Ｆ３：、ＨＤ0：～ＨＤ３：、ＥＭＭ0：～ＥＭＭ９：、
ＭＥＭ0：～ＭＥＭ１：、ＣＯＭ：

ＦＩＬＥＳ、ＳＡＶＥ、ＬＯＡＤなどのファイル入出力コマンド、およびＬＩＳＴ、ＲＵＮなどのコマンドにおいて、デバイス名を省略すると、このコマンドで指定されたデバイス名が指定されたことになります。

〔注意〕ＤＥＶＩＣＥコマンドでデバイス名が指定されていない時にデバイス名を省略した場合、ＢＡＳＩＣ起動時のデバイス名（カセットＢＡＳＩＣ起動時は"ＣＡＳ："、ディスクＢＡＳＩＣ起動時はＢＡＳＩＣを起動させたドライブ）が指定されます。

〔例〕
```
DEVICE"1:"
Ok
LOAD"X1-WORLD"
Ok
```

ロードされるプログラムは、フロッピーディスクのドライブ１にある"Ｘ１－ＷＯＲＬＤ"になります。

1.7 プログラムのマージ

メインメモリ上にあるプログラムと、外部ファイルに記録されたプログラムを合成して、１つのプログラムにすることをマージすると言います。これは、次のＭＥＲＧＥコマンドを実行することによって行なわれます。

```
MERGE"デバイス名:ファイル名"
```

デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

このコマンドが実行されると、指定したデバイスに入っているプログラムが読み込まれ、現在メモリに存在するプログラムと組み合わさった１つのプログラムが作成されます。

〔注意〕双方の行番号が重複した場合は、ファイルから読み込まれた方が優先します。また、セーブされているプログラムはアスキー形式でセーブされている必要があります。アスキー形式でセーブされていないプログラムをマージしようとすると「Bad file mode」のエラーが出ます。

〔例〕①次のプログラムを入力してください。

```
10 DATA 10,20,30,40,50,60,70,80.90,100
20 INPUT N
30 FOR I=1 TO N
40 READ DA(I)
50 NEXT
```

②入力したプログラムをフロッピーディスク（ドライブナンバー0）にアスキー形式でセーブします。

SAVE"0：TEST", A⏎

Ok

＊カセットテープを使用する場合はデバイス名を"0："から"CAS："に変えて実行してください。

＊ＬＩＳＴコマンドによっても行なうことができます。

③メインメモリのプログラムを消します。

ＮＥＷ⏎

Ok

④メインメモリにマージする次のプログラムを入力してください。

```
60 FOR J=1 TO N
70 KEI=KEI+DA(J)
80 NEXT
90 PRINT KEI
```

⑤マージを実行します。

MERGE"0：TEST"⏎

⑥ここでメインメモリのプログラムリストをとります。次のようにマージされた1つのプログラムになっていることが確認できます。

ＬＩＳＴ⏎

```
10 DATA 10,20,30,40,50,60,70,80.90,100
20 INPUT N
30 FOR I=1 TO N
40 READ DA(I)
50 NEXT
60 FOR J=1 TO N
70 KEI=KEI+DA(J)
80 NEXT
90 PRINT KEI
Ok
```

＊このプログラムは行番号１０～５０でＤＡＴＡ文から指定数だけデータを読み込み行番号６０～９０でその合計を出し表示するというものです。

1.8 プログラムのチェイン

プログラムが長すぎて、メインメモリで処理できない場合があります。この場合プログラムをいくつかに分割し、それぞれ別のファイル名で外部ファイルに記録しておきます。そして、1つのプログラム実行後、自動的に次のプログラムをロードして実行するようにします。このことを「プログラムをチェインする」といい次のＣＨＡＩＮコマンドを使用して行います。

CHAIN"〔デバイス名：〕ファイル名"

デバイス名……ＳＡＶＥコマンドと同じものが使えます。

メインメモリ内にあるプログラムの変数を保護し、指定したデバイス内の指定したファイル名のプログラムをロードして実行します。

〔注意〕ロードする前にオープンされていたデータファイルは、このコマンドでは閉じません。

〔例〕サンプル(1)

```
10 DIM IN(20)
20 INPUT"DATA ノ コスウ",N
30 FOR L=1 TO N
40 PRINT L;"バンメノ DATA"
50 INPUT IN(L)
60 NEXT
70 CHAIN"0:SAMPLE2"
```

〔注意〕カセットテープを使用するときは、行番号７０を次のように変えて入力してください。

```
70 CHAIN"CAS:SAMPLE2"
```

サンプル(2)

```
10 FOR M=1 TO N-1
20 B=ABS(IN(M)-IN(M+1))
30 PRINT
40 PRINT IN(M);TAB(10);"-";IN(M+1);TAB(22);"=|";B;"|"
50 NEXT
```

①サンプル(2)のプログラムを入力し、フロッピーディスク（ドライブナンバー０）に"ＳＡＭＰＬＥ２"というファイル名でセーブします。

SAVE"0:SAMPLE2"⏎

Ｏｋ

〔注意〕カセットテープを使用する時は、次のように入力してセーブしてください。

SAVE"CAS:SAMPLE2"⏎

Ｏｋ

②メインメモリをクリアします。

ＮＥＷ⏎

Ｏｋ

③サンプル(1)のプログラムを入力し、実行します。データの個数とデータを入力すると隣り合う２つの数の差をとって、その絶対値が表示されます。

サンプル(1)の行番号１０～６０はデータの入力を行なうだけですが、７０行のＣＨＡＩＮコマンドが実行されることにより、フロッピーディスク（ドライブナンバー０）またはカセットテープからサンプル(2)のプログラムがロードされ実行されます。また、サンプル(2)のプログラムはデータの処理と表示を行なうだけですが、ＣＨＡＩＮコマンドによってサンプル(1)の変数が保護されているため、サンプル(1)のプログラムで入力したデータが処理され表示されます。ただし、ＣＨＡＩＮコマンドが実行されるとサンプル(1)のプログラムは消去されますので注意してください。

1.9 ファイル名の付けかえ

1度セーブしたファイル名を次のNAMEコマンドを実行することによって変更することができます。

```
NAME"〔デバイス名:〕旧ファイル名"AS"〔デバイス名:〕新ファイル名"
```

デバイス名……0:～3:、MEM0:～MEM1:、EMM0:～EMM9:、
F0:～F3:、HD0:～HD3:

旧ファイル名で指定したファイルのファイル名を新ファイル名に変更します。

〔注意〕2つのデバイス名は同じものでなければなりません。また、デバイス名に"CAS:"は使用できません。

〔例〕 **NAME"0:TEST"AS"0:SAMPLE"⏎**
Ok

フロッピーディスク(ドライブナンバー0)の"TEST"というファイルを"SAMPLE"というファイル名に変更します。

1.10 ファイルの属性指定

作成したファイルを誤って消去しないようにそのファイルにライトプロテクト(書き込み禁止)をかけたり、リードアフタライト(ファイルに書き込みを行なう際にデータを書き込んだ後自動的にベリファイを行なう)やシークレット(FILESコマンドよりファイル名の一覧を表示させてもシークレット指定されたファイルは表示されない)というファイルの属性を指定する場合にはSETコマンドを実行します。ただし、SETコマンドはカセットテープに対しては使用することができません。

```
SET { #ファイル番号                    } , { "P" }
    { "〔デバイス名:〕ファイル名"     }   { "R" }
                                         { "S" }
                                         { ""  }
```

ファイル番号……オープンで指定したファイル番号
デバイス名………0:～3:、MEM0:～MEM1:、EMM0:～EMM9:、
F0:～F3:、HD0:～HD3:

属性を指定する文字として「P」と「R」と「S」を使用し、「P」を指定するとライトプロテクト(書き込み禁止)がかかり、「R」を指定するとリードアフタライトが実効され、「S」を指定するとシークレットが実行されます。属性を取り除くにはヌルストリング("")を指定します。

〔例〕

①フロッピーディスク(ドライブ0)内の「SAMPLE」というBASICプログラムファイルにライトプロテクトをかけてみましょう。

SET"0:SAMPLE","P"⏎
Ok

これでライトプロテクトがかかりました。ＦＩＬＥＳコマンドによりファイル名のリストを表示させてみてください。「ＳＡＭＰＬＥ」というファイルのＢasの次に「＊」マークが付いています。このマークがライトプロテクトのかかったファイルであることを示します。

②リードアフタライトの指定は次のようにします。

リードアフタライトを指定するファイルをＯＰＥＮコマンドで開いた後（ファイル番号を１とすると）

ＳＥＴ＃１，"Ｒ"↵

を実行すると､以降このファイルにデータを書き込むと自動的にベリファイ動作を行ないます。このファイルをＣＬＯＳＥコマンドにより閉じた時点でこのリードアフタライトの属性は解除されます。

③例①でライトプロテクトをかけたファイルにさらにシークレット指定を行なってみましょう。

ＳＥＴ"０：ＳＡＭＰＬＥ"，"Ｓ"↵

Ｏｋ

これでシークレット指定されました。ＦＩＬＥＳ"０："↵と入力してファイル名の一覧を表示させてみてください。「ＳＡＭＰＬＥ」というファイル名はどこにも表示されていません。ただし、「ＳＡＭＰＬＥ」というファイルが削除されてしまったわけではなく、単にファイル名の表示を行なわないというだけでそのファイルはディスクに残っています。

1.11 ファイルのパスワード

自分で作成したファイルを他人に使用されたくないという場合には、そのファイルにパスワード（合い言葉）を指定して、データをファイルに書き込みます。すると、このファイルを読み出す時には指定したパスワードを入力しないと読み出せません。このパスワードを指定してファイルのアクセスを行なうには、"**名前〔．エクステンション〕；パスワード**"という形式でファイル名を使用します。

〔注意〕

パスワードには．(ピリオド)"(引用符)：(コロン)；(セミコロン)／(スラッシュ) 記号は使用できません。またファイル名全体の長さは３１文字以内でなければなりません。

〔例〕

たとえば、メインメモリにあるＢＡＳＩＣプログラムをファイルネーム「ＴＥＳＴ」でパスワード「ＡＢＣ」を付けてディスク（ドライブ０）にセーブするには、

ＳＡＶＥ"０：ＴＥＳＴ；ＡＢＣ"↵

と入力します。ＦＩＬＥＳ"０："↵と入力してファイル名のリストを表示してみてください。指定したパスワードは表示されておらず、パスワードを指定せずにセーブしたファイルと表面上は何の変化もありません。しかし、ＬＯＡＤ"０：ＴＥＳＴ"↵と入力して読み出そうとしても「Ｎｏ ｐａｓｓ ｗｏｒｄ」とエラー表示され読み出すことができません。読み出すためにはＬＯＡＤ"０：ＴＥＳＴ；ＡＢＣ"↵と入力します。これにより、指定したパスワードを知らない人はそのファイルを扱うことができなくなります。

2 データファイル

プログラム中で使用するデータは、外部ファイルにデータファイルという形で再生・保存が可能です。また、データファイルにはシーケンシャルアクセスファイルとランダムアクセスファイルの２種類があります。

シーケンシャルアクセスファイルとは、連続的に書かれたデータファイルのことで、次のような特徴があります。

1）ファイルの中では、データは書き込まれた順序で並んでおり、データを保存・再生する際には先頭から順番にしか行なえません。

2）データの内容を変更する際には、内容を変更しないデータも順に読み出し、処理を行なった後、ファイルに書き出すことが必要です。

3）記録したファイルの最後にデータを追加することができます。

4）ファイルの中のデータは順序よく並んでおり、無駄がありません。

5）ファイルの作成が比較的簡単に行なえます。

シーケンシャルアクセスファイルを使用してデータの保存・再生を行なう場合、次のコマンドを使用します。

OPEN、PRINT#、WRITE#、INPUT#、LINPUT#、CLOSE、EOF、INPUT$、DEVICE、OPTION SCREEN、INIT MAXFILES

これに対してランダムアクセスファイルとは、１レコード（＝２５６バイト）単位でその記録位置を指定できるデータファイルのことで、次のような特徴があります。

1）ファイル中のデータは、内部のインデックスによって直接その記憶装置を指定できるようになっているため、データの保存・再生はレコードごとにランダムに行なえます。

2）ファイル中の一部のデータだけを変更することができます。

3）記録したファイルの最後にデータを追加することができます。

4）データが少ない場合でもロード、セーブを１レコードごとに行なうため無駄な部分があります。

5）ファイルの作成のために多くの設定が必要なため、シーケシャンアクセス ファイルに比べてプログラムが複雑になります。

ランダムアクセスファイルを使用してデータの保存・再生を行なう場合、次のコマンドを使用します。

OPEN、CLOSE、FIELD、LSET、RSET、GET、PUT、DEVICE、OPTION SCREEN、INIT、MKI$、MKS$、MKD$、CVI、CVS、CVD

シーケンシャルアクセスファイルを使えば、ほとんどのデータ処理が行なえますが、ランダムファイルを自由に使いこなしてこそ、初めてコンピュータの価値が発揮できます。

〔注意〕データファイルもプログラムファイルと同様にＫＩＬＬ、ＦＩＬＥＳ、ＬＦＩＬＦＳ、ＮＡＭＥ、ＳＥＴ命令によりファイルの削除、ファイル名の一覧表示等が行なえます。

2.1 シーケンシャルアクセスファイルへのデータの保存・再生

シーケンシャルアクセスファイルへのデータの保存・再生を行なう場合、プログラムファイルのように1つのコマンドだけで行なうことはできません。シーケンシャルアクセスファイルの場合書き込み、読み出しのコマンド（ＰＲＩＮＴ＃、ＩＮＰＵＴ＃など）の前後でファイルの状態を設定する必要があります。ここでは、シーケンシャルアクセスファイルへのデータの保存・再生について、実際のプログラム例を用いて説明します。なお、コマンドの詳細に関しては『ＢＡＳＩＣリファレンスマニュアル』を参照してください。

次のプログラムを入力してください。

```
10 OPEN"O",#1,"1:DATA"
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE
40 PRINT#1,NA$
50 PRINT#1,AGE
60 CLOSE#1
100 OPEN"I",#1,"1:DATA"
110 INPUT#1,NA$
120 INPUT#1,AGE
130 PRINT"名前は";NA$"です。"
140 PRINT"歳は";AGE;"才です。"
150 CLOSE#1
```

〔注意〕カセットテープを使用する場合は、行番号１０および１００を次のように書き換え、さらに行番号７０を新たに入力してください。

```
10 OPEN"O",#1,"CAS:DATA"
70 APSS-1
100 OPEN"I",#1,"CAS:DATA"
```

入力がおわりましたら、プログラムのリストをとり、間違いがないかどうかを調べてください。

それでは、プログラムの説明を行ないます。

```
10 OPEN"O",#1,"1:DATA"
```

シーケンシャルアクセスファイルへデータを書き込んだり、読み出したりするには、どのファイルにどのような操作を行なうのかを宣言する必要があります。その宣言を行なうのがこのＯＰＥＮコマンドで、各パラメータによって次のことが設定されます。

"Ｏ"……ファイルへのデータの書き込み（ＯＵＴＰＵＴ）を設定します。

＃１……ＯＰＥＮコマンドによって出力するファイル番号を設定します。

"１：ＤＡＴＡ"……出力デバイスを設定します。

すなわち、行番号１０の実行でフロッピーディスク（ドライブナンバー１）にデータの出力が可能になります。また、以後のファイル処理は、ファイル名ではなくファイル番号で行ないます。

＊この設定を行なうことをファイルを開く、あるいはファイルをオープンするといいます。

```
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE
```

行番号２０、３０でシーケンシャルアクセスファイルに出力するデータを用意します。

```
40 PRINT#1,NA$
50 PRINT#1,AGE
```

行番号４０、５０では、ファイル番号１でオープンされたファイルにデータを出力します。シーケ

ンシャルアクセスファイルにデータを出力するコマンドは、このＰＲＩＮＴ＃の他にＷＲＩＴＥ＃があります。この２つのコマンドの違いについては『ＢＡＳＩＣリファレンスマニュアル』を参照してください。

```
60 CLOSE#1
```

すべてのデータを書き込んだならば、その内容を保持するためにオープンしたファイルをとじるという処理が必要です。そのためのコマンドがＣＬＯＳＥで、行番号６０ではファイル番号１のファイルを閉じています。このようにして閉じられたファイルは、再びオープンされない限り読み出すことも書き込むこともできなくなります。以上でシーケンシャルアクセスファイルへのデータの出力は終りです。

```
100 OPEN"I",#1,"1:DATA"
```

次に、書き込んだデータを読み出します。読み出す場合も書き込む場合と同様にＯＰＥＮ文でファイルをオープンする必要があります。ただ、書き込みを指定していたパラメータ"Ｏ"を"Ｉ"に変え、読み出し（ＩＮＰＵＴ）状態に設定する必要があります。行番号１００の実行によって、フロッピーディスク(ドライブナンバー１)のシーケンシャルアクセスファイル"ＤＡＴＡ"からデータの読み出しが可能になります。

```
110 INPUT#1,NA$
120 INPUT#1,AGE
```

行番号１１０、１２０ではファイル番号１でオープンされたファイルからデータを変数ＮＡ＄、ＡＧＥに読み込みます。シーケンシャルアクセスファイルでは、データを読み出す場合、書き込んだ順番に行なう必要があり、一部のデータを読み出す場合でも、その前に書き込まれたデータをいったん読み出さなければなりません。

```
130 PRINT"名前は";NA$"です。"
140 PRINT"歳は";AGE;"才です。"
```

行番号１３０、１４０では、シーケンシャルアクセスファイルから読み込んだデータを画面に表示します。

```
150 CLOSE#1
```

行番号１５０ではファイル番号１のファイルを閉じています。データ書き込みだけでなく、読み込みの場合でも、その実行が終ったならばファイルを閉じる必要があります。

また、カセットテープを使用する場合、行番号７０を追加しました。

```
70 APSS-1
```

これは、行番号１０から６０まで作成したファイルの頭出しを行なうためのステートメントです。

〔注意〕データの保存、再生を行なう場合、プログラムの保存、再生と同様にディスクならば、フォーマットが、グラフィックメモリならばＯＰＴＩＯＮ　ＳＣＲＥＥＮ、ＩＮＩＴが実行されている必要があります。

〔例〕では、実際の実行例を示します。

入力する文字「パソコンテレビ」は全角文字、「２」は半角文字で入力してください。
＊プログラムの実行前にＫＭＯＤＥ１が実行されていない場合は漢字を表示しません。

2.2 シーケンシャルアクセスファイルへのデータの追加

シーケンシャルアクセスファイルでは、ファイルの任意の位置への書き込み、読み出しは行なえませんが、ファイルの一番最後にデータを追加することはできます。ただし、カセットテープではデータの追加ができません。ここでは、２．１のプログラムで作られたシーケンシャルアクセスファイルにデータを追加する例を用いて説明します。

次のプログラムを入力してください。

＊なお、このプログラムは２．１のプログラムの一部を変更することで簡単に作れます。

```
10 OPEN"A",#1,"1:DATA"
20 INPUT"名前は";NA$
25 IF NA$="END"THEN 60
30 INPUT"何才ですか";AGE
40 PRINT#1,NA$
50 PRINT#1,AGE
55 GOTO 20
60 CLOSE#1
100 OPEN"I",#1,"1:DATA"
105 IF EOF(1) THEN 150
110 INPUT#1,NA$
120 INPUT#1,AGE
130 PRINT"名前は";NA$"です。"
140 PRINT"歳は";AGE;"才です。"
145 GOTO 105
150 CLOSE#1
```

それでは、プログラムの説明を行ないます。

```
10 OPEN"A",#1,"1:DATA"
```

データの追加でファイルをオープンする場合、パラメータを"Ａ"にしデータの追加（ＡＰＰＥＮＤ）状態にしなければなりません。行番号１０ではフロッピーディスク（ドライブナンバー１）の"ＤＡＴＡ"というシーケンシャルアクセスファイルにデータを追加することを宣言しています。

```
20 INPUT"名前は";NA$
25 IF NA$="END"THEN 60
30 INPUT"何才ですか";AGE
40 PRINT#1,NA$
50 PRINT#1,AGE
55 GOTO 20
60 CLOSE#1
```

行番号２５のＩＦ文と行番号５５のＧＯＴＯ文で何回もデータが入力できるようになっています。シーケンシャルアクセスファイルにデータを追加する場合ＯＰＥＮコマンドのパラメータ以外は、入力と同じ手続きで行ないます。追加されるデータは、それ以前のデータの後に入力順に追加されます。行番号６０まででデータの追加は終りです。

```
100 OPEN"I",#1,"1:DATA"
105 IF EOF(1) THEN 150
110 INPUT#1,NA$
120 INPUT#1,AGE
130 PRINT"名前は";NA$"です。"
140 PRINT"歳は";AGE;"才です。"
145 GOTO 105
150 CLOSE#1
```

行番号１００～１５０では、追加されたシーケンシャルアクセスファイルからデータの入力を行ない画面に表示します。行番号１０５のＩＦ文はシーケンシャルアクセスファイルの終りをＥＯＦ

（END　OF　FILEの略）関数により検出し、終わりであれば行番号150へジャンプすることを意味します。EOF関数は、最後のデータならば真の値（－1）を与え、そうでないならば偽の値（0）を与えます（詳しくは、『BASICリファレンスマニュアル』を参照してください。）
〔例〕以下は、プログラムの実行例です。

＊「名前は？」の所でENDをキー入力することによってデータの追加が終わります。

2.3 ランダムアクセスファイルへのデータの保存・再生

ランダムアクセスファイルもシーケンシャルアクセスファイルと同様に書き込み、読み出しのコマンド前後でファイルの状態を設定する必要があります。また、ランダムファイルは、データの入出力をファイルバッファ（メインメモリ上でデータを一時的に蓄える部分）を介して行ない、そのファイルバッファへの入出力は文字型のデータでなければなりません。そのためシーケンシャルアクセスファイルよりも次のように多くの手続きを必要とします。

1．データの保存

①ファイルをオープンします。……………………………………〔OPEN〕
②ランダムアクセスファイルのデータを用意します。データが数値型ならば文字型に変更を行なう必要があります。〔MKI$、MKS$、MKD$〕
③ファイルバッファに文字変数を割りつけます。……………〔FIELD〕
④ファイルバッファにデータを書き込みます。………………〔LSET、RSET〕
⑤ファイルバッファの内容をディスク、グラフィックメモリに書き込みます。…………………………………………………〔PUT〕
⑥ファイルを閉じます。………………………………………………〔CLOSE〕

2．データの再生

①ファイルをオープンします。……………………………………〔OPEN〕
②ファイルバッファに文字2変数を割りつけます。……………〔FIELD〕
③ランダムアクセスファイル上のデータをファイルバッファに読み込みます。……………………………………………〔GET〕

④ファイルバッファの上のデータを取り出して処理します。
文字型に変換していた数値をもとの数値型に変換しなおします。……………………………………〔CVI、CVS、CVD〕
⑤ファイルを閉じます。……………………………〔CLOSE〕
〔注意〕カセットテープでは、ランダムアクセスファイルを扱うことはできません。

それでは、実際のプログラム例を用いて説明していきます。なお、各コマンドの詳細に関しては『ＢＡＳＩＣリファレンスマニュアル』を参照してください。
次のプログラムを入力してください。

プログラム(1)………データ保存例

```
10 OPEN"R",#1,"1:RDATA"
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE%
40 T$=MKI$(AGE%)
50 FIELD#1,10 AS A$,2 AS B$
60 LSET A$=NA$
70 LSET B$=T$
80 PUT#1,1
90 CLOSE#1
```

それでは、プログラムの説明を行ないます。

```
10 OPEN"R",#1,"1:RDATA"
```

ランダムアクセスファイルをオープンする場合"Ｒ"のモード（Ｒandam file mode）指定を行ないます。なお、ランダムファイルの場合入力、出力で別のモード指定を行なう必要はなく１つのＯＰＥＮ文でデータの入出力をすることも可能です。行番号１０では、フロッピーディスク（ドライブナンバー１）に"ＲＤＡＴＡ"という名のランダムアクセスファイルをオープンしています。

```
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE%
40 T$=MKI$(AGE%)
```

ランダムアクセスファイルに出力するデータは文字型のものだけで、数値型のものは使用することができません。そのため、数式を文字列に変換した後、出力する必要があります。数式を文字列に変換するコマンドは、行番号４０のＭＫＩ＄の他にＭＫＳ＄、ＭＫＤ＄があります。それぞれのコマンドに関しては『ＢＡＳＣリファレンスマニュアル』を参照してください。

```
50 FIELD#1,10 AS A$,2 AS B$
```

ランダムアクセスファイルにデータの入出力を行なう場合、１レコード（２５６バイト）のデータをファイルバッファに蓄えておき、１レコード分ずつ入出力を行ないます。また、そのファイルバッファは、入出力の前にデータの割り付けがなされていなければなりません。それを行なうのがＦＩＥＬＤ文で行番号５０の場合、ファイル番号１用のファイルバッファ（２５６バイト）の最初の１０バイトをＡ＄、次の２バイトをＢ＄に割り付けています。なおこの場合、残りの２４４バイトは未使用となり、たいへんぜいたくな使い方といえます。
＊ＦＩＥＬＤ文はファイルバッファに変数の割り付けを行うだけで、データをファイルバッファおよびファイルに入出力しません。

```
60 LSET A$=NA$
70 LSET B$=T$
```

行番号６０、７０ではＦＩＥＬＤ文で割り付けたファイルバッファにデータを出力しています。ファイルバッファにデータを出力する場合、このＬＳＥＴの他にＲＳＥＴというコマンドがあります。２つの違いについては『ＢＡＳＩＣリファレンスマニュアル』を参照してください。

```
80 PUT#1,1
```

ＰＵＴ文によってファイルバッファの内容（２５６バイト）がランダムアクセスファイルに出力されます。行番号８０では、ファイル番号１のファイルバッファにあるデータをＯＰＥＮ文で指定した"ＲＤＡＴＡ"というファイルのレコード番号１に書き込んでいます。ランダムアクセスファイルには、レコード番号が記録されるようになっており、このレコード番号の指定を変えることによりデータのランダムな入出力が可能となります。

```
90 CLOSE#1
```

行番号９０では、オープンされていたランダムアクセスファイルを閉じています。ランダムアクセスファイルの場合、入出力別にＯＰＥＮ文を実行する必要がないため、ファイルを閉じなければ、データの入出力を１つのＯＰＥＮ文、ＦＩＥＬＤ文によって行なうことができます。

〔例〕実行例を以下に示します。

＊入力する文字「パソコンテレビ」は全角文字で入力してください。

次にランダムアクセスファイルからのデータ入力について説明します。

次のプログラムを入力してください。

プログラム(2)………データ再生例

```
10 OPEN"R",#1,"1:RDATA"
20 FIELD#1,10 AS A$,2 AS B$
30 GET#1,1
40 AGE=CVI(B$)
50 PRINT    "名前は";A$;"です。"
60 PRINT    "歳は";AGE;"才です。"
70 CLOSE#1
```

それでは、プログラムの説明を行ないます。

```
10 OPEN"R",#1,"1:RDATA"
20 FIELD#1,10 AS A$,2 AS B$
```

行番号１０でランダムアクセスファイルをデータ入力のためにオープンしています。ランダムアクセスファイルの場合、前に述べたようにデータの入出力でオープン方法は変わりません。行番号２０では、ファイルバッファに変数を割り付けています。この場合、出力時と変数名はかえることができますが、ファイルバッファの割り付けは同じにする必要があります。

```
30 GET#1,1
```

ファイルから１レコード（２５６バイト）分のデータをファイルバッファに入力するにはＧＥＴ文を使います。行番号３０の場合、ファイル番号１のファイルバッファにＯＰＥＮ文で指定した"ＲＤＡＴＡ"というファイルのレコード番号１からデータを２５６バイト読み込んでいます。

```
40 AGE=CVI(B$)
50 PRINT   "名前は";A$;"です。"
60 PRINT   "歳は";AGE;"才です。"
70 CLOSE#1
```

ファイルバッファに読み込まれたデータは、ＦＩＥＬＤ文で割り付けた文字変数を使用することによって取り出せます。また、一旦文字式に変換した数値は、ＣＶＩ、ＣＶＳ、ＣＶＤコマンドを実行することによりもとの数値に変換することができます。３つのコマンドの違いについては『ＢＡＳＩＣリファレンスマニュアル』を参照してください。以上のことは行番号４０～６０で行なわれており、行番号７０では入力のためにオープンしたファイルを閉じています。

〔例〕それでは実行例を示します。なおこのプログラムの実行前にプログラム(1)が実行されている必要があります。

＊名前の入力をする際、全角文字でパソコンテレビと入力しましたが、ＦＩＥＬＤ文でそのファイルバッファの領域を１０バイトしか設けなかったため、このプログラムでは"パソコンテ"としか表示されません。データをファイルバッファに割り付ける場合、その大きさに注意する必要があります。

2.4 ランダムアクセスファイルに関するデータの追加・変更

ランダムアクセスファイルはシーケンシャルアクセスファイルに比べて多くの手続きを必要としますが、データの追加・変更は自由に行なうことができます。すなわち、レコード番号を指定することによって、どの場所のデータもアクセス（呼び出し）可能です。ここでは、２．３で作成したランダムアクセスファイルにデータを追加・変換するプログラム例を説明します。

次のプログラムを入力してください。

プログラム(3)………データ追加・変換プログラム

```
10 OPEN"R",#1,"1:RDATA"
20 INPUT"名前は";NA$
25 IF NA$="END"THEN 90
30 INPUT"何才ですか";AGE%
40 T$=MKI$(AGE%)
50 FIELD#1,10 AS A$,2 AS B$
60 LSET A$=NA$
70 LSET B$=T$
75 INPUT"レコード番号";N
80 PUT#1,N
85 GOTO 20
90 CLOSE#1
```

ほとんどがプログラム(1)の場合と同じです。行番号７５で入力するレコード番号を変えることにより、どの場所のデータでも追加変更できます。なお、このプログラムでは名前に「ＥＮＤ」を入力するまで何回でも追加・変更を繰り返すことができます。
〔例〕実行例を示します。

それでは、プログラム(3)で作成したランダムファイルからデータを取り出す次のプログラムを入力してください。
プログラム(4)

```
10 OPEN"R",#1,"1:RDATA"
20 FIELD#1,10 AS A$,2 AS B$
25 INPUT"レコード番号";N
28 IF N=0 THEN 70
30 GET#1,N
40 AGE=CVI(B$)
50 PRINT"名前は";A$;"です。"
60 PRINT"歳は";AGE;"才です。"
65 GOTO 25
70 CLOSE#1
```

この場合もまた、プログラム(2)とほとんど同じです。行番号２５でアクセスするレコード番号を入力し、行番号６５によって繰り返しが行なえるようになっています。
〔例〕実行例を示します。

＊ファイル番号に０を入力することによってプログラムの実行が終ります。
また、「ＲＤＡＴＡ」というファイルのレコード番号１には、プログラム(1)の実行により「パソコンテ」というデータが入っていましたが、プログラム(3)の実行例で示すようにレコード番号１に新しく「小林」というデータを書き込んだため、前のデータは消去されています。

9章

ミュージック

本機には、プログラムサウンドゼネレータ（ＰＳＧ）という、擬似音発生用ＬＳＩが内蔵されています。このＰＳＧを使うことにより音楽演奏やゲームの効果音を作ることができます。ここではこのＰＳＧをコントロールするサウンド制御ステートメントの使い方を説明します。

なお、サウンド制御ステートメントには次のようなものがあります。

BEEP, MUSIC, PLAY, TEMPO, SOUND, SOUND@, MUSIC@, PLAY@

1　ブザー音を出す

ブザー音を出すためにはＢＥＥＰステートメントを次のように使用します。

BEEP	**0………ブザー音を止める。** **1………ブザー音を出し続ける。** **省略……短い「ポッ」という音を出す。**

9章

〔例〕

```
BEEP 1 ⏎
Ok
BEEP 0 ⏎
Ok
```

2　楽譜を演奏する

ＭＵＳＩＣ、ＰＬＡＹ、ＴＥＭＰＯのステートメントを使用して楽譜の演奏（８オクターブ、３重和音）が行なえます。

MUSIC 文字式

文字式で指定された楽譜に従って演奏します。文字式では、音程、長さ、休符、オクターブ、音量、和音の指定を以下のようにして行なうことができます。

(1)**音程**……………音程は下記のようにC～Bの文字を使用して指定します。

音程	ド	ド# (レb)	レ	レ# (ミb)	ミ	ファ	ファ# (ソb)	ソ	ソ# (ラb)	ラ	ラ# (シb)	シ
指定文字	C	#C	D	#D	E	F	#F	G	#G	A	#A	B

(2)**長さ**……………音程の後ろに0～9の整数をつけて、音の長さを指定します。

*音の長さは4分音符（整数5）を1としたときの相対的な値です。

また、整数の指定がない場合は前の音と同じ長さを意味します。

音の長さ	対応する整数
$\frac{1}{8}$（32分音符）	0
$\frac{1}{4}$（16分音符）	1
$\frac{3}{8}$（付点16分音符）	2
$\frac{1}{2}$（8分音符）	3
$\frac{3}{4}$（付点8分音符）	4
1（4分音符）	5
$1\frac{1}{2}$（付点4分音符）	6
2（2分音符）	7
3（付点2分音符）	8
4（全音符）	9

③休符……………R0～R9で、休みの長さを指定します。
*数字の値は上記の長さに対応しています。

④オクターブ……"O1"～"O8"のようにアルファベットの"O"の後ろに1～8の整数をつけてオクターブの指定をします。また、音符の前に+をつけることによって上の音程を、－をつけることによって下の音程を指定できます。初期状態はO4です。

⑤音量……………V0～V15で音量の指定をします。なお、この指定の次から音量が変化します。

⑥和音……………2重音や3重和音を演奏するには、上声部と下声部の間にチャンネルセパレータ":"（コロン）をはさみます。
"チャンネルA：チャンネルB：チャンネルC"

TEMPO　数値（30～7500）

MUSIC文によって演奏される曲の速さ（テンポ）を指定します。

*指定された数値は1分間あたりの4分音符の拍数を示します。
初期状態では120に設定されています。

PLAY　{数値 | 文字式}

MUSIC文とTEMPO文の両方の機能を持ち、PLAYの後ろが数式の場合TEMPO文と同様の、文字式の場合MUSIC文と同様の機能を持ちます。

〔例〕オクターブを1から8まで変化させてからテンポを速くし、これを繰返します。

```
10 'MUSIC or PLAY
20 WIDTH 40:INIT
30 CLS4
40 PLAY"V15R0:V15R0:V15R0" ――――――――――――音量最大　休符は最小。
50 FOR T=120 TO 2000 STEP 100 ――――――――テンポを変化させる。
60 TEMPO T
70 FOR O=1 TO 8 ―――――――――――――――――オクターブを1から8まで変化させる。
80 PLAY"O"+CHR$(48+O)+"CDEFGABAGFEDC:DEFGABAGFEDCB:EFGABAGFEDCBA"
90 NEXT O
100 NEXT T
```

ＭＵＳＩＣ＠、ＰＬＡＹ＠ステートメントは音の長さをＣＴＣでコントロールしていますので音を出しながら、グラフィックステートメント等を実行することができます。ただしプログラムの中でのみ使用可能です。

3　ＰＳＧのコントロール

本機で音を出すには、先に述べたＰＬＡＹステートメント等によって楽譜を文字列にして演奏する方法とＳＯＵＮＤ、ＳＯＵＮＤ＠を使用して直接ＰＳＧにデータを送り音を出す方法があります。ここでは、ＳＯＵＮＤ、ＳＯＵＮＤ＠の使い方を説明します。

SOUND　PSGのレジスタ番号, データ〔, データ……〕

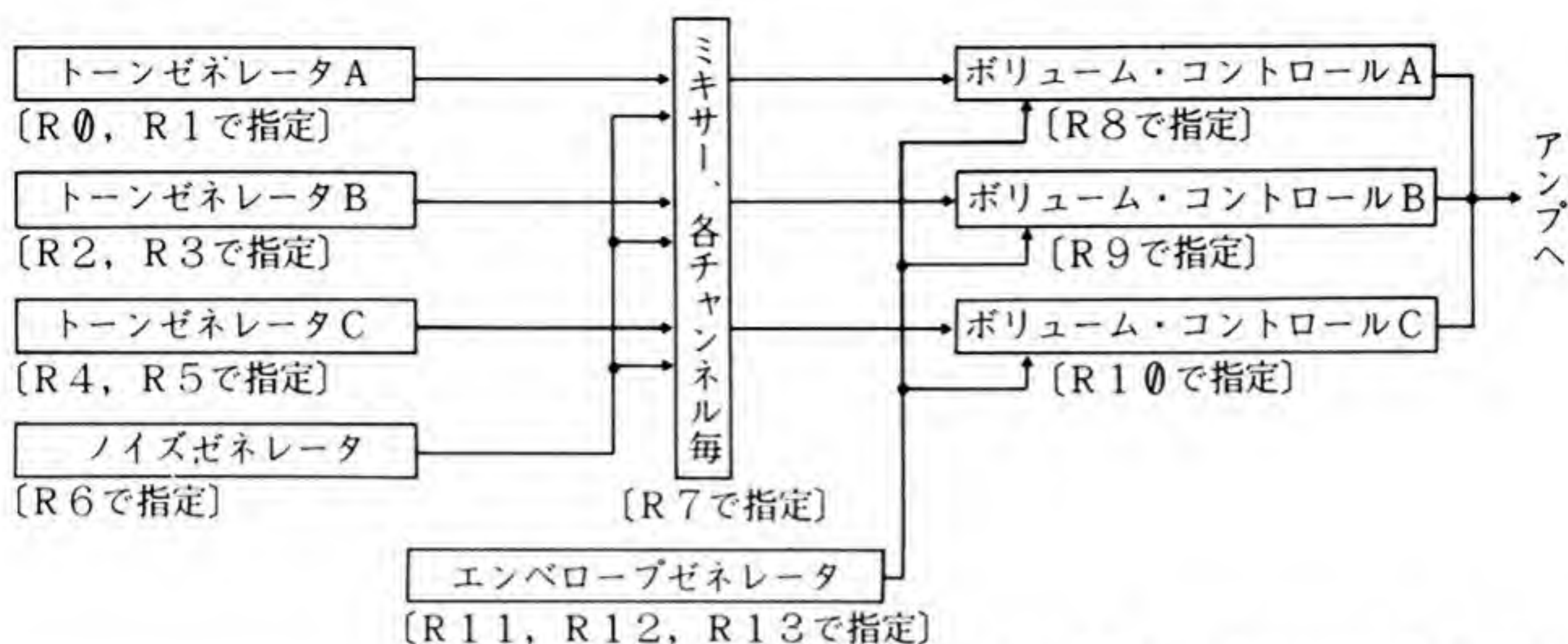
SOUND@　PSGのレジスタ番号, データ〔, データ……〕

ＳＯＵＮＤ、ＳＯＵＮＤ＠命令は、ＰＳＧの１４個のレジスタへデータを書き込んで、ゲームの効果音など様々な音を作り出す事ができます。

なお、ＳＯＵＮＤ命令は、１つのデータで１つのレジスタ（８ビット）を定義しますが、ＳＯＵＮＤ＠命令は、１つのデータで連続したレジスタ（１６ビット）を定義できます。後で説明する周波数の設定のように連続したレジスタを定義する場合にＳＯＵＮＤ＠を使用します。

また、データをカンマ（,）で区切って並べて書くと、連続した複数のレジスタにそのデータを書き込むことができます。

本機に内蔵しているＰＳＧは、３つのトーンゼネレータ、ノイズゼネレータ、エンベロープゼネレータ、ミキサー、３つのボリュームコントロールから構成されています。

そのブロック図を次に示します。

（注）図中のR0～R13はレジスタ番号です

上のブロックのレジスタにデータを送ることによりさまざまな音が発生します。
その各レジスタの機能表を以下に示します。

レジスタ番号	レジスタ機能	ビット構成 7	6	5	4	3	2	1	0	設定データ値（15進）番号	
0	チャンネルA周波数	下位8ビットT_L（微調）								0（高）～255（低）	
1						上位4ビットT_H（粗調）				0（高）～15（低）	
2	チャンネルB周波数	下位8ビットT_L（微調）								0（高）～255（低）	
3						上位4ビットT_H（粗調）				0（高）～15（低）	
4	チャンネルC周波数	下位8ビットT_L（微調）								0（高）～255（低）	
5						上位4ビットT_H（粗調）				0（高）～15（低）	
6	ノイズ周波数				5ビットN					0（高）～31（低）	
7	チャンネル選択	$\overline{IN}$/OUT		ノイズ			トーン			0～63	
		IOB	IOA	C	B	A	C	B	A	（各ビット共、0を設定すると選択される）	
8	チャンネルA音量				M	4ビット				0(小)～16(大)	各チャンネルとも M＝0のとき（データ値0～15）4ビットの値で0から15段階に音量設定可能 M＝1のとき（データ値16～31）音量はエンベロープに依存し4ビットデータは無効になる。
9	チャンネルB音量				M	4ビット				0(小)～16(大)	
10	チャンネルC音量				M	4ビット				0(小)～16(大)	
11	エンベロープ周期	下位8ビットE_L（微調）								0（小）～255（大）	
12		上位8ビットE_H（粗調）								0（小）～255（大）	
13	エンベロープ形状					4ビット				0～15	

*レジスタ番号7のビット6、7は、SOUND、SOUND@命令では使用しません。

それでは各ブロックの設定方法を説明します。

(1) トーンゼネレータ

レジスタ0～5では、A、B、C各チャンネルの周波数を決定します。周波数は音の音程にあた

り、１オクターブ音が上ると周波数は倍になります。また、その間を１２に分けると半音階が得られます。
下の表は、オクターブ４の各音の周波数を表わしています。

音　階	C	＃C	D	＃D	E
周波数（Ｈｚ）	２６１．６３	２７７．１８	２９３．６６	３１１．１３	３２９．６３

音　階	F	＃F	G	＃G	A
周波数（Ｈｚ）	３４９．２３	３６９．９９	３９２．００	４１５．３０	４４０．００

音　階	＃A	B
周波数（Ｈｚ）	４６６．１６	４９３．８８

＊音階はＭＵＳＩＣの文字列で表わしています。
ＰＳＧでは入力周波数である２ＭＨｚを内部で１６分の１に分周（周波数を分割する操作）したものを、各チャンネルの１２ビットデータでさらに分周して発生させます。
求めたい周波数をｆ〔Ｈｚ〕、分周値をＴ〔Ｈｚ〕とすると次の関係があります。

$$f=\frac{2\times10^6}{16\times T}\cdots\cdots\cdots\cdots\cdots①$$

たとえば、１ＫＨｚの音を発生するには、①を変形した次の式に周波数を代入することにより分周値を求めることができます。

$$T=\frac{2\times10^6}{16\times f}\cdots\cdots\cdots\cdots\cdots②$$

代入すると

$$T=\frac{2\times10^6}{16\times10^3}=125$$

この分周値を各チャンネルのレジスタ（Ａチャンネルの場合、レジスタ１の上位４ビットとレジスタ中の下位８ビット）に指定するところにより１ＫＨｚの音が設定できます。
Ａチャンネルに１ＫＨｚを設定する場合次のようにします。

```
SOUND@0, 125
```

＊この場合レジスタ０には１２５が、レジスタ１には０が設定されます。

また、４４０Ｈｚ（Ｏ４Ａ）を指定したい場合は次のようにします。

$$T=\frac{2\times10^6}{16\times440}\fallingdotseq284$$

チャンネルＢに４４０Ｈｚ（Ｏ４Ａ）を指定する場合、次のようにします。

```
SOUND@2, 284
```

＊この場合レジスタ３には１が、レジスタ２には２８が送られます。

(2) ノイズゼネレータ

レジスタ6によって雑音の平均周波数を指定します。雑音とは周波数が不規則に変化するものを指し、レジスタ6の5ビットの値によって分周値を設定します。

たとえば、5KHzのノイズを発生するには②の式に周波数を代入し分周波を求めます。

$$T=\frac{2\times10^6}{16\times5000}=25$$

この値をレジスタに設定する場合1つのレジスタのみを指定すればよいためSOUNDステートメントを使います。

SOUND　6，25

(3) ミキサー（チャンネル選択）

レジスタ7では、A、B、C各チャンネル毎にトーン／ノイズ／（トーン＋ノイズ）出力の有無を選択します。たとえば、

SOUND7，44 または **SOUND7，&B1Ø11ØØ**

と設定すると、Aチャンネルからはトーンだけ、Bチャンネルからはトーン＋ノイズ、Cチャンネルからはどちらも出力されません。

(4) ボリュームコントロール

レジスタ番号8～1Øは、各チャンネルの音量を決定します。ただし、Mビット（5ビット目）の値により、4ビットデータによるØ～15までの一定した音量が設定できるモード（M＝Ø）と、自動的に音量がØ～15まで変化するエンベロープモード（M＝1）とを選択できます。

たとえば、

SOUND　9，12

と設定するとBチャンネルの音量を12に設定できます。

(5) エンベロープゼネレータ

レジスタ11，12では、エンベロープの周波数を指定し、レジスタ13ではエンベロープの形状を指定します。

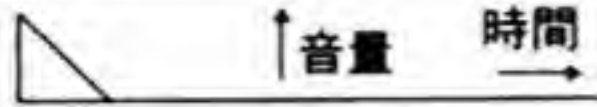
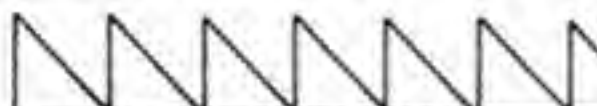
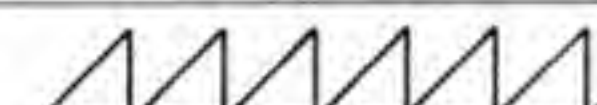
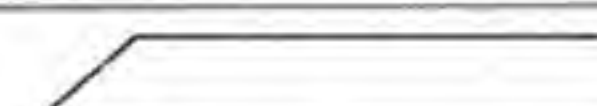
まず、周波数の設定方法を説明します。PSGでは、エンベロープの周波数は入力周波数2MHzを256分の1に分周し、それをレジスタ11、12の16ビットの値でさらに分周しています。そのため、エンベロープ周波数をf_E〔Hz〕、分周値をTとすると次の関係があります。

$$f_E=\frac{2\times10^6}{256\times T}\cdots\cdots\cdots③$$

たとえば、f_E＝Ø．5〔Hz〕（エンベロープ周期1／f_E＝2〔秒〕）を設定するには、③式を変形した次の式に周波数を代入することにより分周値を求めることができます。

$$T=\frac{2\times10^6}{256\times f_E}\cdots\cdots\cdots④$$

代入すると

$$T=\frac{2\times10^6}{256\times0.5}=15625$$

この分周値を次のように指定することによってØ．5〔Hz〕が設定できます。

SOUND@11，15625

＊この場合、レジスタ１１に９が、レジスタ１２に６１が設定されます。

次にエンベロープの形状を決定します。

それには、下の表に示すパターンに対応したデータをレジスタ１３に指定することが必要です。

R13のデータ	エンベロープパターン
0～3	↑音量　時間→
4～7	
8	
9	
10	
11	
12	
13	
14	
15	

|←→| 1/fE＝エンベロープ周期

〔例〕では、実際にＳＯＵＮＤ，ＳＯＵＮＤ＠ステートメントで音を出してみます。下のプログラムを実行してください。

```
10 SOUND@0,284         ……………Aチャンネルの周波数設定。
20 SOUND6,25           ……………ノイズ周波数設定。
30 SOUND@11,15625      ……………エンベロープ周期の設定。
40 SOUND13,8           ……………エンベロープパターンの設定。
50 SOUND8,15           ……………Aチャンネルの音量設定。
60 SOUND7,&B111110     ……………Aチャンネルの選択。
```

Aチャンネルから４４０Ｈｚ（Ｏ４Ａ）の音が出ます。音をとめるには CTRL ＋ D を押してください。次に６０行目も次のように再入力してください。

```
60 SOUND7,&B110110     ……………Aチャンネルのノイズとトーンを選択。
```

Aチャンネルからは、音とノイズの両方が出力されます。次に５０行目を次のように再入力してください。

```
50 SOUND8,16           ……………エンベロープモードを設定。
```

Aチャンネルの音量にエンベロープがかかり、おもしろい音になります。

また、ＳＯＵＮＤステートメントでエンベロープを指定することにより、ＰＬＡＹ（ＭＵＳＩＣ）ステートメントによる同一音の演奏を音符毎に区切ることができます。

```
PLAY120:PLAY"V16O4C5CDDEED"
```

上のＰＬＡＹステートメントを、次のように変えてみてください。ただし、文字式での音量指定（Ｖ）はＶ１６にします。

```
10 SOUND@11,5245:SOUND13,0:SOUND8,16
20 PLAY120:PLAY"V16O4C5CDDEED"
```

10章 配列

コンピュータで扱うデータがプログラムの実行によって変化するような場合、**変数**というものを使います。

変数はプログラムで使われるデータの内容により**数値変数**とか**文字変数**と呼びます。

データの数が少ない場合には使われる変数の数は少なくてすみますが、１００個、１０００個…というようにデータの数が増えると、いくつもの変数を用意しなくてはならず、プログラムが複雑になってしまいますし、メモリのムダも多くなります。

そこで、このように扱うデータ量が多くなった場合には、配列を使用すると便利です。

配列というのは一組のデータに番号を付けて規則的に並べたもので、以下に例をあげて説明します。

たとえば、郵便屋さんが、あるアパートに手紙を配達する場合を考えてみてください。そして、そのアパートが5階建てで、各階に１０件ずつの家があるとします。

さて、ここで、各家庭に手紙を配達しようとしたとき、手紙の宛名をみながら一軒一軒手紙を配っていくと、階段を何度も上ったり降りたりして、たいへん効率の悪い配り方となります。ところが、この手紙は2階の3軒目の家の手紙というように、それぞれの手紙を何階の何軒目という具合に整理してから配達すると、時間も短くて済み、効率が良く配達できます。

コンピュータでデータを扱う場合も同様で、データの数が増えるほど一組にまとめて、番号順に並べて整理をした方が、扱いやすいものとなります。

このようにデータに一連の番号を付けて、並べたものを**配列**といいます。

1　配列名と添字

配列は変数の集まりですので名前を付けて扱います。この名前を**配列名**といい、使用文字や長さは一般の変数と同様です。

配列で使用する一連の番号は(　)でくくり、その配列上の位置を示しており**添字**と呼びます。
つまり配列データを参照する場合は、配列名と添字を指定します。
また、配列の個々のデータを**配列要素**といいます。

配列データの書き方は次のようにします。

配列名（添字）

たとえばAという配列名で、5個のデータを格納する場合次のようになります。

A(5)

A(1)	A(2)	A(3)	A(4)	A(5)

さらに、添字はカンマ(,)で区切ってB（3，3）のように並べることができます。
この例では、3×3＝9個のデータを格納できます。

B（3，3）

B（1，1）	B（1，2）	B（1，3）
B（2，1）	B（2，2）	B（2，3）
B（3，1）	B（3，2）	B（3，3）

B（3，3）とした場合、添字が2個ありますので、**2次元配列**といいます。
添字が3個ある場合には**3次元配列**、n個ある場合は**n次元配列**といいます。
次元は一行で扱える文字数との関係から125次元程度まで作成できます。

2 配列宣言

配列を使用する場合は、あらかじめ使用するデータ領域を確保する必要があります。
これを**配列宣言**といい**DIMステートメント**を使います。
DIMステートメントは次のように書きます。

```
DIM 配列名（大きさ、大きさ、………）
```

ここで大きさと書いたのは添字の最大値のことで、この値によって指定された大きさの記憶場所を確保します。
また配列宣言は省略することも可能です。
配列宣言を省略して配列変数を使った場合は

DIM 配列名（10）

と宣言したのと同じになります。

3 OPTION BASE

配列の添字の最小値は0か1です。
この値を決めるには **OPTION BASEステートメント**を使用します。

```
OPTION BASE { 0 }
            { 1 }
```

BASIC起動時には、添字の最小値は0になっていますがOPTION BASE1を宣言することによって添字の最小値を1に設定することができます。また、1度OPTION BASEステートメントを宣言すると再度宣言することができません。
OPTION BASEを実行する場合は配列変数があらかじめすべて消去されている必要があります。
もし配列変数が存在した状態で OPTION BASEステートメントを実行しようとするとDuplicated definitionのエラーとなります。

4　配列変数の消去

ＤＩＭステートメントで宣言した配列を消去するには**ＣＬＥＡＲステートメント**もしくは、**ＥＲＡＳＥステートメント**を使用します。

ＣＬＥＡＲステートメントを実行すると、変数はすべて消去されますが、このとき、配列変数もすべて消去されます。

ＥＲＡＳＥステートメントは指定した配列変数の内容のみ消去し、次のように使います。

ＥＲＡＳＥ　配列名、配列名、………

5　ＶＤＩＭ

ＤＩＭステートメントで宣言される配列変数は、メインメモリ上に記憶領域を確保します。

したがって、大量の配列を宣言すると、プログラムテキストとして使用する部分が少なくなり、Ｏｕｔ　ｏｆ　ｍｅｍｏｒｙエラーの原因となります。

そこで、大量のデータを扱う場合には、グラフィックＶＲＡＭの一部を変数領域として確保し、メインメモリにプログラムテキストを書く方法が便利です。

この場合、ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　１　もしくは２で、グラフィックＶＲＡＭを、変数エリアとして使用するようにあらかじめ定義しておきます。

前記のスクリーンモードで、グラフィックＶＲＡＭに配列変数（もしくは通常の変数）領域を確保する命令が**ＶＤＩＭステートメント**です。

ＶＤＩＭステートメントは

ＶＤＩＭ　配列名（大きさ、大きさ……）

のように書きます。

ＶＤＩＭステートメントで宣言された配列変数は記憶領域がグラフィックＶＲＡＭに確保されること以外はＤＩＭステートメントで宣言された配列変数と同様の使い方をします。

ただし、ＶＤＩＭステートメントで宣言された配列変数はＣＬＥＡＲステートメントでは消去されません。

消去するには、ＶＤＩＭ　ＣＬＥＡＲというステートメントを用います。

また、ＥＲＡＳＥステートメントはＤＩＭの場合と同様に、ＶＤＩＭで宣言された配列変数を消去することができます。

〔配列を使用したサンプルプログラム〕

```
10 WIDTH 40,25,0
20 INIT:CLS
30 OPTION SCREEN1
100 DIM A$(255,1)
110 FOR I=0 TO 255
115 CLS:LOCATE 0,0:PRINT HEX$(I)
120 FOR J=0 TO 1
130 A$(I,J)=CGPAT$((J+1)*256+I)
150 PRINT
160 PRINT#0,A$(I,J):PAUSE 5
170 NEXT J
180 NEXT I
```

⇦配列データをメインメモリ上ではなくグラフィックＲＡＭ上にとる場合は、１００行のＤＩＭ　１００行のＤＩＭ　Ａ＄（２５５，１）をＶＤＩＭ　Ａ＄（２５５，１）に変更します。

この例は２次元配列を利用したプログラムです。

１００行で、Ａ＄という配列を宣言しています。１３０行でキャラクタゼネレータのデータを読み出し、配列の中に入れています。配列　Ａ＄（Ｉ，Ｊ）のうち　Ｊ＝０ときＲＯＭのデータが入り、Ｊ＝１のときＲＡＭのデータが入ります。１６０行で読み出した内容を表示しています。

以上の操作を２５６回くり返しています。

配列データをメインメモリ上ではなくグラフィックＶＲＡＭ上にとる場合は、１００行のＤＩＭ　Ａ＄（２５５，１）をＶＤＩＭ　Ａ＄（２５５，１）に変更します。

11章

機械語サブルーチンとモニタ

ここでは、ＢＡＳＩＣプログラムを機械語サブルーチンとともに使用する方法を説明します。

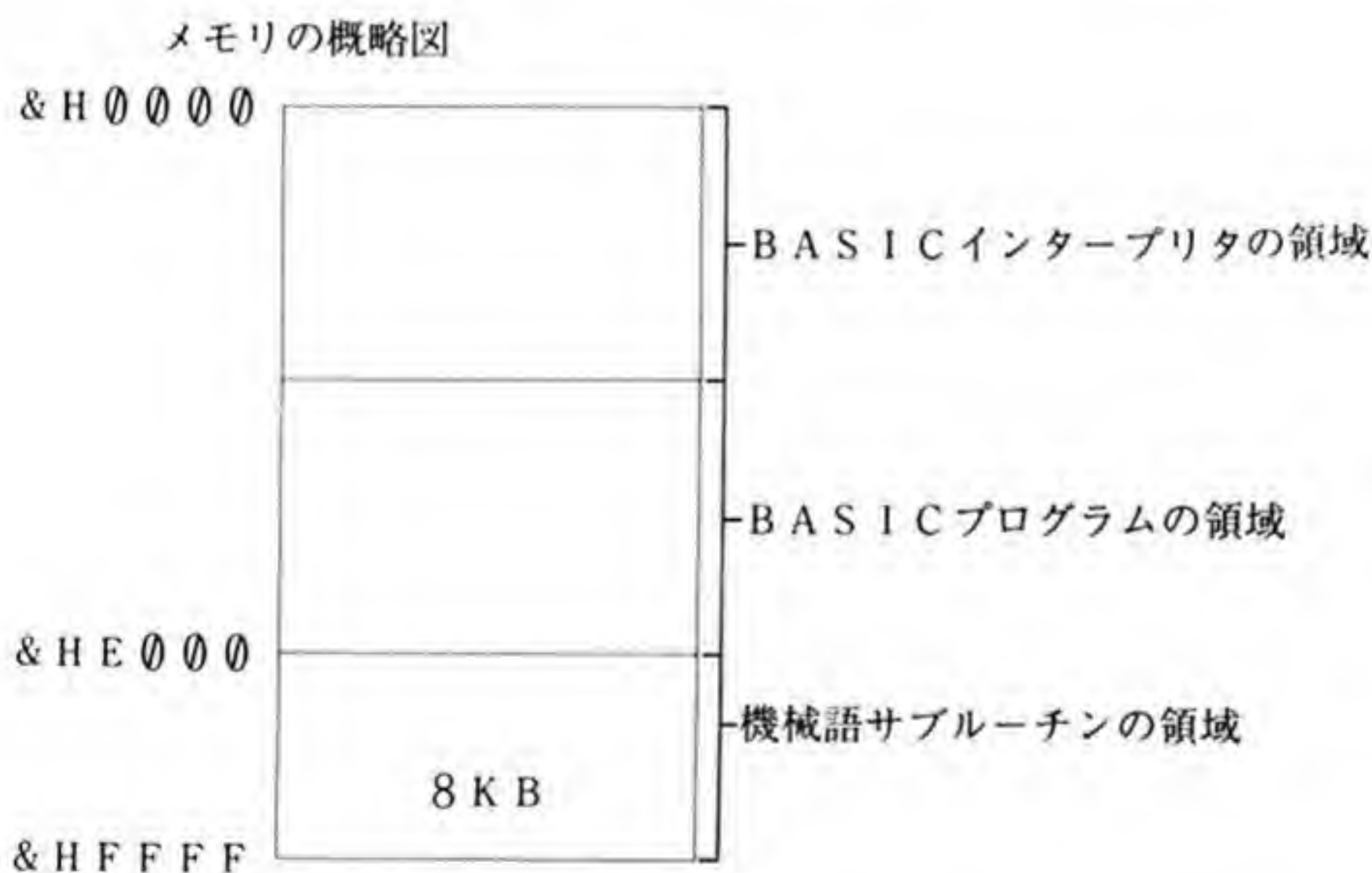
1　機械語サブルーチンの入力

1.1　機械語サブルーチンの記憶領域の設定

機械語サブルーチンを入力するためには、まずＣＬＥＡＲコマンドを用いて、その記憶領域を確保する必要があります。

たとえば、ＢＡＳＩＣの６４ＫＢのメインメモリ中、上部の８ＫＢ（最上部の３ＫＢはワークエリアとして使用されるので実際は５ＫＢ）を確保したければ、

CLEAR　&HE000

というステートメントをプログラムの先頭に書くか、そのままダイレクトに実行します。

これで、ＢＡＳＩＣプログラムが機械語サブルーチンの領域に侵入することがなくなります。

ただし、最上部の＆ＨＦ４００～＆ＨＦＦＦＦの３Ｋバイトは、ＢＡＳＩＣ、ＩＰＬ（Ｉｎｉｔｉａｌ　Ｐｒｏｇｒａｍ　Ｌｏａｄｅｒ）およびＢＩＯＳのワークエリアとなっているため使用することができません。また、＆ＨＦ０００～＆ＨＦ３ＦＦは辞書のエリアとして使用されるので、辞書を使うときはこのエリアを使用することができません。

11章

1.2 機械語サブルーチンの入力方法

機械語サブルーチンをメモリに記憶させる方法には、次の２通りがあります。

(1) ＰＯＫＥステートメントを使う。

(2) ＭＯＮコマンドで機械語モニタを起動する。

(1) ＰＯＫＥステートメントによる入力

ＰＯＫＥステートメントを使って、機械語サブルーチンを入力する手順を次に示します。

1．機械語（１６進数表現のコード）によってプログラムを作ります。

2．機械語を、１バイトごとにカンマ（,）で区切って、ＤＡＴＡ文の中にデータとして用意します。

3．機械語データをＲＥＡＤステートメントで逐次読み込んで、ＰＯＫＥステートメントでメモリに書き込みます。

この方法は、比較的小さなプログラムを記憶するときに使います。

次に簡単な例を示します。

```
100 'POKEステートメントによる入力
110 CLEAR &HD000
120 I=0
130 READ D$
140 D=VAL("&H"+D$)
150 POKE &HD000+I,D
160 IF D=&HC9 THEN 190
170 I=I+1
180 GOTO 130
190 FOR J=&HD000 TO &HD000+I
200 RD=PEEK(J)
210 PRINT HEX$(RD)
220 NEXT
230 END
240 '機械語データ
250 DATA 3E,07      :'LD   A,07H
260 DATA 01,00,20 :'LD   BC,2000H
270 DATA ED,79      :'OUT  (C),A
280 DATA 3E,41      :'LD   A,41H
290 DATA 01,00,30 :'LD   BC,3000H
300 DATA ED,79      :'OUT  (C),A
310 DATA C9         :'RET
```

このプログラムは、＆ＨＤ０００番地から機械語データを入力しています。

１１０：機械語サブルーチンの記憶領域を＆ＨＤ０００から確保しています。

１３０～１８０：２５０～３１０のＤＡＴＡ文で指定された機械語データを読み
ＰＯＫＥステートメントでメモリに書き込んでいます。

１９０～２２０：書き込んだデータをＰＥＥＫ関数で読み出して表示しています。

(2)機械語モニタによる入力

機械語モニタ（後述「③機械語モニタ」参照）を用いて、機械語サブルーチンを入力する方法を次に示します。

1．ＭＯＮ⏎と打ち込んで、機械語モニタを起動します。

2．モニタのＤコマンドかＭコマンドを実行し、メモリに機械語（１６進数表現のコード）を書き込みます。

3．必要に応じてＳコマンドを使い、作った機械語サブルーチンをカセットテープに記録します。

4．Ｒコマンドで機械語モニタからＢＡＳＩＣに戻ります。

2　BASICプログラムから機械語サブルーチンを呼び出す方法

ＢＡＳＩＣから機械語サブルーチンを呼び出すには、ＣＡＬＬステートメントを使う方法と、ＵＳＲ関数を使う方法の２通りがあります。

2.1 CALLステートメント

機械語サブルーチンは、ＢＡＳＩＣのＣＡＬＬステートメントを使って呼び出すことができます。ＣＡＬＬステートメントの形式は次の通りです。

```
CALL a
```

a：機械語サブルーチンの開始アドレス。

機械語サブルーチンの最後には、もとのプログラムに戻るためにＣ９（１６進数、ニーモニックではＲＥＴ）が置かれます。

ＣＡＬＬステートメントを実行すると、プログラム・カウンタの内容をスタックに退避し、ＣＡＬＬステートメント中に指示されたアドレスａ、すなわちサブルーチンの入口へジャンプします。

機械語サブルーチンの処理を終えてＣ９に来ると、スタックに退避しておいたプログラムカウンタをもとに戻し、もとのＢＡＳＩＣプログラムの実行を再開します。

ＣＡＬＬステートメントではＢＡＳＩＣ上のデータを機械語サブルーチンへ引き渡す機能がありません。もし、データの引き渡しが必要な場合には次に述べるＵＳＲ関数を使用する方法が便利です。

2.2 USR関数

ＵＳＲ関数は、引数の値を受け渡すことができ、また、エラー処理のサブルーチンを呼び出すことができる、という利点を備えています。

ＵＳＲ関数を使うためには、まず、呼び出す機械語サブルーチンの実行開始アドレスを定義する必要があります。それにはＤＥＦＵＳＲステートメントを使って、次のように書きます。

```
DEFUSRn=a
```

n：ＵＳＲ関数識別番号。０～９の整数。

a：実行開始アドレス。

（例）

```
100 DEFUSR0=&HE000
110 DEFUSR1=&HEC00
```

nは、関数識別番号で、０～９の最大１０個の機械語サブルーチンを定義することができます。nを省略すると０が指定されます。

実行開始アドレスａは、機械語サブルーチンの実行が開始されるアドレスであり、メイン・メモリ６４ＫＢ中のどこにでも指定できます。

ＤＥＦＵＳＲでいったんアドレスを定義すると、再度定義しなおすまで、そのアドレスが保持されます。

ＤＥＦＵＳＲで、ｎとａを定義したら次のようにしてＵＳＲ関数を使うことができます。

```
〔1〕 y＝USRn(x)
〔2〕 y$＝USRn(x$)
```

〔1〕 ｙ：数値変数。
ｎ：ＵＳＲ関数識別番号。0～9の整数。
ｘ：数式（引数）。

〔2〕 ｙ＄：文字変数。
ｎ：ＵＳＲ関数識別番号。0～9の整数。
ｘ＄：文字式（引数）。

（例）
```
200 A=USR0(65)
210 A$=USR1(C$)
```

ＵＳＲの後ろにつける番号ｎは、ＤＥＦＵＳＲで設定した番号に対応しています。

ＵＳＲ関数は、引数ｘやｘ＄の値をもって、ｎ番の機械語サブルーチンを呼び出し、リターン時にその値をｙやｙ＄に代入します。機械語サブルーチンの中で、引数の値を書き換えると、その結果がｙやｙ＄の値となります。

ＵＳＲ関数の引数ｘ、ｘ＄は省略することができません。

ｘは、数式で、単独の定数、数値変数でもかまわず、精度も整数型、単精度型、倍精度型のいずれでもかまいません。

ｘ＄は、文字列を値にもつ文字式です。

ＵＳＲ関数で機械語サブルーチンを呼びだすとき、ＣＰＵの各レジスタには次のような値が入っています。

●Ａレジスタ（アキュムレータ）

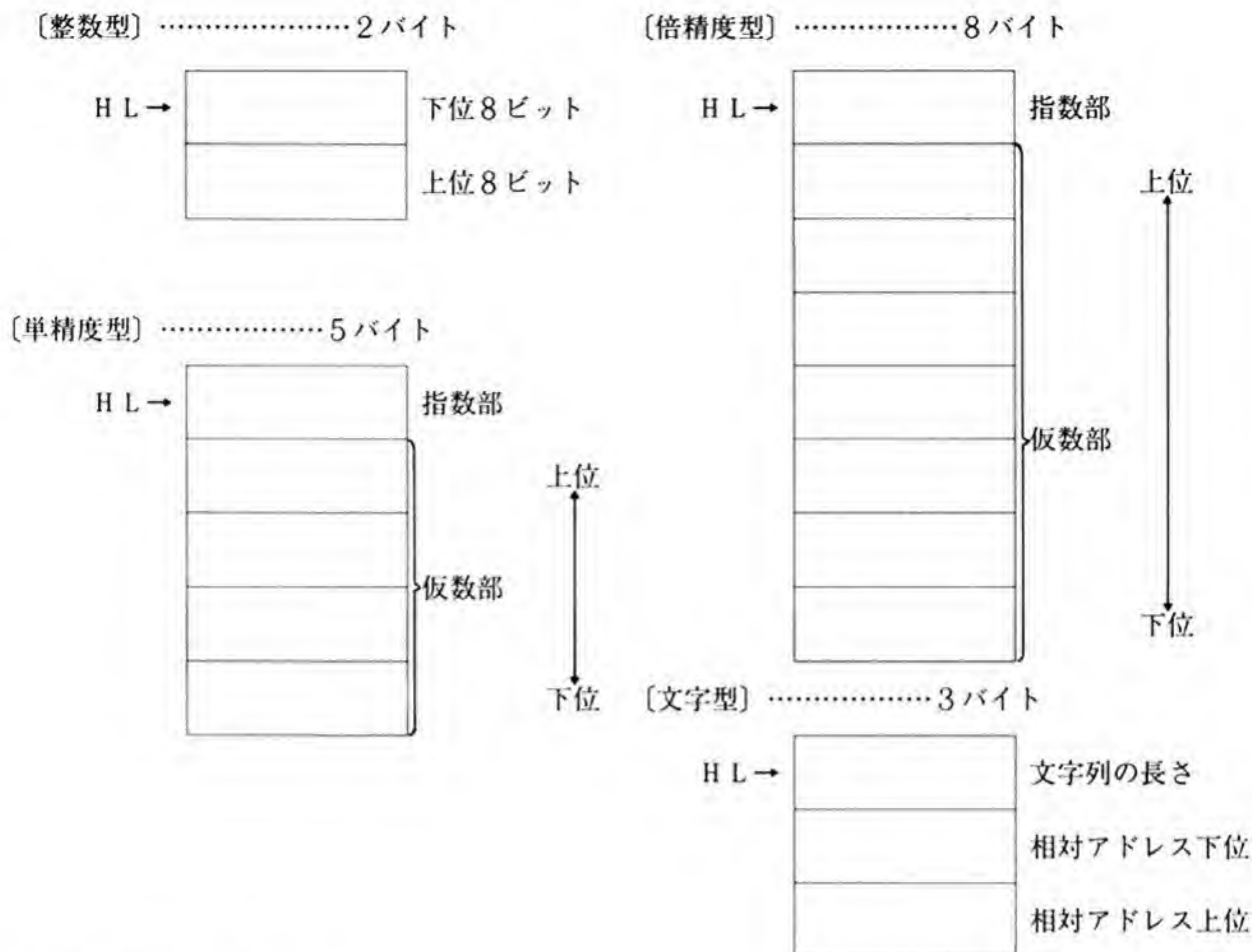
Ａレジスタ（アキュムレータ）には、引数の型を示す値が入っています。

引数の型	Ａレジスタの値
整数型	2
単精度型	5
倍精度型	8
文字型	3

これらの値は、文字変数を除き、メモリ内において何バイトで表現されているかを表わしています。

●ＨＬレジスタ

ＨＬレジスタの値は、数値変数と文字変数の場合で異なっています。数値変数の場合は引数の入っているメイン・メモリのアドレスを示しており、引数が文字変数の場合は、直接文字列の入っているアドレスを示さず、ストリング・ディスクリプタと呼ばれている相対アドレスを示しています。ストリング・ディスクプリタは、メモリの文字変数領域の先頭アドレスから何番地目に文字列が入っているのかを示す値で、直接文字列の入っているアドレスを示すものではありません。文字列の入っているアドレスは、ＤＥレジスタに入っています。

引数の値は、ＨＬレジスタが指しているアドレスから次のような形式で格納されています。

●DEおよびBレジスタ

DEとBレジスタは、Aレジスタが3のとき、すなわちUSR関数の引数が文字変数のときのみ意味をもちます。

このとき、DEレジスタにはその文字データが格納されているアドレス（絶対アドレス）、Bレジスタには文字データの長さが入っています。

USR関数の引数が文字変数だけのときは、引数の格納されているアドレスとUSR関数実行後の引数のアドレスが同じなので、引数の値が変わってしまう可能性があります。したがって、文字変数を引数にするときは、

USR（A$+""）　　　※""はヌルストリングです。

というように、ヌルストリングを連結した形にするようにしてください。これで、引数A$の値がUSR関数実行前と実行後とで変わることはなくなります。

●USR関数内でのエラー処理

USR関数の中でエラーが発生した場合、BASICにエラーの発生をしらせることができます。このとき、BASIC内であらかじめON　ERROR　GOTOの処理をしていれば、エラー処理ルーチンへジャンプすることもできます。

その方法は、エラー番号をAレジスタに入れIXレジスタの示すアドレスにジャンプすることにより行なうことができます。

(例) 機械語サブルーチン中にエラーが発生したら、Type mismatch (エラー番号13) の処理へジャンプする場合、

```
        :
        :
        :
      JP   Z, TYPEMS………エラーが発生したら分岐します。
        :
        :
        :
      RET
TYPEMS : LD   A, 13……………エラー番号13をAレジスタへいれます。IXレジ
         JP  (IX)           スタの示すアドレスへジャンプします。
```

3 機械語モニタ

本機は機械語の扱いを容易にするため、簡単な機械語モニタを備えています。
機械語モニタを起動するには、BASICの**MONコマンド**をダイレクトに入力してください。

```
:
Ok
MON⏎
*
```

MONコマンドを入力すると、上のように*印が表示され、機械語モニタが起動します。

また、本機の電源をONにするかBASICでBOOT⏎を実行して、IPLを起動しているとき SHIFT + BREAK を押しつづけると、次の画面となります。
この画面のとき、M キーを押しても機械語モニタを起動することができます。ただし、この場合はモニタのコマンドのRで機械語モニタから抜け出ることができません。

```
          Make your device ready

      Press selected key to start driving:

                F : Floppy
                R : ROM
                C : CMT
                T : Timer
```

機械語モニタのコマンド

機械語モニタには、次に示す14種類のコマンド用意されています。

コマンド	働き
D	Dump memory（メモリダンプ）
M	set Memory（メモリセット）
F	Find data（データサーチ）
P	Printer switch（プリンタスイッチ）
G	Gosub
T	Transfer data（データ転送）
S	Save
L	Load
V	Verify
R	Return
!	BIOS ROM／MAIN RAM access mode switch
#	WIDTH40／80 switch、画面の初期化など
W	Write（各デバイスへ）
Y	read（各デバイスから）

D　（dump memory）

働き：メインメモリの内容を表示します。

文法：＊D〔XXXX〔YYYY〕〕

XXXX：先頭アドレス。16進数4けた以内。0～FFFF。

YYYY：最終アドレス。16進数4けた以内。0～FFFF。

説明：XXXXからYYYYまでのメインメモリの内容を表示します。（例として、40文字モード時の表示について示します）

YYYYを省略すると、XXXXから128バイト分（8バイト×16行）表示します。XXXXもいっしょに省略すると、それまで表示した次のアドレスから128バイト分表示します。

画面の1行分の表示形式は、次のようになります。

```
:XXXX=HH HH HH HH HH HH HH HH/CCCCC.C.
↑ ↑  ↑             ↑          ↑       ↑
(1)(2)(3)          (4)        (5)     (6)
```

(1)　編集可能なことを示すマーク。

(2)　先頭データのアドレス。16進数4けた。0000～FFFF

(3) アドレスとデータ群の区切り記号。

(4) １６進数のデータ８バイト。ＨＨが００～ＦＦの１６進数で、空白をはさんで８個または１６個並ぶ。

(5) データ群と対応文字の区切り記号。

(6) (4)のデータ群に対応する文字列。ＨＨをキャラクタ・コード（１６進数）とする文字を表示します。ＨＨがコントロール・コードのときは、ピリオド(.)が表示されます。

また、８０文字モード時には、１行１６バイト分のデータが表示され、ＹＹＹＹを省略すると２５６バイト分のデータが一度に表示されます。

この状態のとき、次の〔編集書式〕に従って、メインメモリの内容を書き換えることができます。

表示のストップは BREAK キー、再開はスペースキーか ⏎ キーを押します。また、コマンド待ちに戻るには SHIFT ＋ BREAK キーを押します。

〔編集書式〕

機械語モニタもＢＡＳＩＣと同様にスクリーン・エディタを備えており、メインメモリのダンプ後、あるいはサーチ後、次の書式に従って内容を書き換えることができます。

〔書式〕　：アドレス＝データ　データ　…………………
```
  ↑     ↑   ↑   ↑   ↑                ↑   ↑
 (1)   (2) (3) (4) (5)              (4) (5)
```

(1) コロン：ＢＡＳＩＣの行番号と(2)のアドレスを間違えないための記号で、編集可能な行であることを示します。

(2) アドレス：先頭データのアドレス。１６進数４けたで指定します。００００～ＦＦＦＦ。

(3) イコール：アドレスとデータ群の区切り記号。

(4) データ：メモリに書き込まれるデータで、次の４通りの指定の仕方があります。

a) ＨＨ⇨１６進数２けた。

b) ；Ｘ⇨Ｘは１文字。

c) "ＸＸ………Ｘ"

d) 'ＸＸ………Ｘ'

a) の場合、１６進数２けたの値が書き込まれます。

b) の場合、セミコロン(;)に続く１バイトに対応するキャラクタ・コードが書き込まれます。

c) または d) の場合、ＸＸ………Ｘの部分にメッセージを書くことができます。

ただし、c) の場合、メッセージ中に引用符（"）を含めることができません。

また、d) の場合、メーセージ中にアポストロフィ（'）を含めることができません。

メッセージは行の先頭から入力してください。

(例)　：８０００＝"ＣＡＮ'Ｔ　ＣＯＮＴＩＮＵＥ"

　　　：９０００＝'ＬＯＡＤＩＮＧ"ＢＡＳＩＣ"'

(5) 空白：データとデータの間は、原則として、空白で区切ります。

書き換えた後、その行で ⏎ キーを押すと、その行のデータがメモリに記憶されます。

M (set memory)

働き：メインメモリの内容を1バイト表示します。

文法：＊M〔 XXXX〕

XXXX：先頭アドレス。16進数4けた以内。0～FFFF。

説明：MXXXX⏎と入力すると、そのアドレスの内容を表示して、データの入力待ちとなります。

```
＊MFE00⏎
：FE00=[H]H
        ↑
      カーソル
```

このとき、前述の〔編集書式〕に従って、データを書き換えることができます。書き換えた後、⏎キーを押すと、データがメモリに記憶され、記憶されたデータ数から次のアドレスが計算されて、改行されます。

```
＊MFE00⏎
：FE00=3E 41 13 00 C9⏎
：FE06=[H]H
        ↑
      カーソル
```

この状態から抜け出すには、SHIFT + BREAK キーを押します。

F (find data)

働き： データを探し出します。

文法： ＊F XXXX YYYY HH HH ……………………

XXXX：先頭アドレス。16進数4けた以内。0～FFFF。

YYYY：最終アドレス。16進数4けた以内。0～FFFF。

HH：データ。16進数2けた以内。0～FF。

説明： XXXXからYYYYまでのメインメモリから、HH HH …………で指定された一続きのデータ群を探し出し、見つかったときは、そのアドレスとデータを表示します。

```
＊F1000 2000 CD 41 00⏎
：10C3=CD 41 00 ／ヘA.
＊■
 ↑
カーソル
```

上の例は、「CD 41 00」という3バイトのデータを、メモリの1000～2000番地（16進数）から探して、10C3番地にそれが見つかったことを示しています。

このコマンドを実行すると、該当するアドレスとそのデータを、あるだけ何行でも表示します。

途中で、このコマンドから抜け出たいときは、［SHIFT］＋［BREAK］キーを押してください。

P（printer switch）

働き：　画面表示とプリンタ表示の切り換えをします。

文法：　＊P

説明：　Pコマンドを実行すると、DおよびFコマンド実行後の表示を、プリンタにしたり、画面にしたり、切り換えることができます。

機械語モニタを起動した時点では、画面に表示するようになっていますが、このコマンドを入力するたびに、プリンタ表示／画面表示、と表示するデバイスが反転します。

プリンタが正しく接続されていない場合は、しばらくしてから「Ｅｒｒ　７３」と画面に表示されてコマンド待ちになります。そのときは、プリンタをチェックするか、再度Ｐコマンドを実行して表示を画面に戻してください。

G（gosub）

働き：　機械語サブルーチンを呼び出します。

文法：　＊G　HHHH

ＨＨＨＨ：　アドレス。１６進数４けた以内。０～ＦＦＦＦ

説明：　メインメモリ上のＨＨＨから始まるサブルーチンを呼び出します。

T（transfer data）

働き：　データの転送をします。

文法：　＊T　XXXX　YYYY　ZZZZ

ＸＸＸＸ：　データの先頭アドレス。１６進数４けた以内。０～ＦＦＦＦ

ＹＹＹＹ：　データの最終アドレス。１６進数４けた以内。０～ＦＦＦＦ

ＺＺＺＺ：　データの転送先のアドレス。１６進数４けた以内。０～ＦＦＦＦ

説明：　メインメモリ上のＸＸＸＸからＹＹＹＹまでのデータをＺＺＺＺを先頭とする場所に転送します。

S（save）

働き：　メインメモリの内容をカセット・テープに記録します。

文法：　＊S　XXXX　YYYY　ZZZZ:file name

ＸＸＸＸ：　先頭アドレス。１６進数４けた以内。０～ＦＦＦＦ

ＹＹＹＹ：　最終アドレス。１６進数４けた以内。０～ＦＦＦＦ

ＺＺＺＺ：　実行開始アドレス。１６進数４けた以内。０～ＦＦＦＦ

説明：　ＸＸＸＸからＹＹＹＹまでのメインメモリの内容をカセットテープに記録します。

再びＬコマンドでロードするとき、先頭アドレスが省略されると、ＸＸＸＸからロードされます。

ＺＺＺＺは、ＩＰＬから機械語プログラムを走らせたり、ＢＡＳＩＣからＬＯＡＤＭコマンドのＲオプションによって走らせるときに、実行の開始アドレスとなります。

(1) ロード時の先頭アドレス
(2) ロード時の最終アドレス
(3) 実行開始アドレス
(4) ファイル名
file nameは、13文字以内のファイル名と3文字以内のエクステンションからなり、:の後ろに続けて書きます。

L (load)

働き： カセットのデータをメインメモリに読み込みます。

文法： *L〔XXXX〕〔:file name〕

XXXX： ロード先頭アドレス。16進数4けた以内。0〜FFFF

説明： カセット・テープに記録されているデータや機械語プログラムをメインメモリに読み込みます。

XXXXを指定すると、そのアドレスから読み込み、省略すると、Sコマンドで記録した通りの形で読み込みます。

file nameを省略すると、最初に見つけたファイルを読み込みます。

読み込んでいる途中で、SHIFT + BREAK キーを押したり、その他のエラーが生じたときは、「Err 29」と表示されコマンド待ちの状態に戻ります。
このコマンドは機械語プログラムを読み込むだけで実行は行ないません。
このコマンドによって、読み込んだ機械語プログラムを実行するには、Gコマンドを使います。

V (verify)

働き： カセットの内容とメインメモリの内容を比較します。

文法： *V〔:file name〕

説明： 指定したファイルをカセット・テープから読み込み、メインメモリの内容と比較します。

Sコマンドで機械語プログラムやデータが正しく記録されたかどうか調べるときに使います。もしカセット・テープから読み込んだデータとメモリの内容と一致していなければ、「Err 29」を出して止まります。

「Err 29」が出たときは、再度Sコマンドで記録しなおしてください。

R (return)

働き： 機械語モニタからシステムに戻ります。

文法： *R

説明： 機械語モニタを起動したBASICに戻ります。

このとき、SP(スタック・ポインタ)およびHLレジスタの内容は保存されているので、MONの次のコマンドやステートメントに移ります。次の命令がないときは、コマンド待ちの状態となります。

```
*R
Ok
■
↑
カーソル
```

!（change access mode）

働き：　ＩＰＬ　ＲＯＭとメインメモリのアクセスの切り換えをします。

文法：　＊！

　　　　）！

説明：　モニタのコマンド（Ｄ、Ｍ、Ｆ、Ｇ、Ｔ）でアクセスされるメモリ、バンクをＩＰＬ　ＲＯＭにするか、メインメモリにするかを切り換えることができます。

　機械語モニタを起動した時点では、メインメモリがアクセスされており、コマンド待ちのとき「＊」が表示されますが、

　　＊！↵

を実行すると、ＩＰＬ　ＲＯＭがアクセスされるようになり、コマンド待ちのとき「）」が表示されます。

　また、その状態で

　　）！

を実行すると、「＊」が表示されメインメモリのアクセスに戻ります。

　ＩＰＬ　ＲＯＭのアクセス時、Ｄコマンドによる読み込むメモリの表示はＩＰＬ　ＲＯＭになっていますが､書き込むメモリはあくまでもメインメモリになるので注意してください。

#（set screen mode）

働き：　画面モードの切り換え、ＰＡＬＥＴの初期化、グラフィック画面の消去をします。

文法：　〔1〕　＊＃

　　　　〔2〕　＊＃Ｐ

　　　　〔3〕　＊＃Ｃ

　　　　〔4〕　＊＃ｋｄｍ

　　　　　　　ｋ：　0または1。

　　　　　　　ｄ：　0～2。

　　　　　　　ｍ：　0～5の整数。

説明：　〔1〕　＊＃↵

　　　　　　ＷＩＤＴＨ４0とＷＩＤＴＨ８0の切り換えをします。

　　　〔2〕　＊＃Ｐ↵

　　　　　　ＰＡＬＥＴの初期化をします。

　　　　　　　ＰＡＬＥＴ0，0：ＰＡＬＥＴ1，1：ＰＡＬＥＴ2，2
　　　　　　：ＰＡＬＥＴ3，3：ＰＡＬＥＴ4，4：ＰＡＬＥＴ5，5
　　　　　　：ＰＡＬＥＴ6，6：ＰＡＬＥＴ7，7↵またはＰＡＬＥＴ@0，1，2，
　　　　　　　3，4，5，6，7↵と同じ

　　　〔3〕　＊＃Ｃ↵

　　　　　　グラフィック画面の消去をします。

　　　　　　　（＝ＣＬＳ0↵）

〔4〕 ＊＃ｋｄｍ↵

k、d、m の値によって次のような設定ができます。

k の値	コ ー ド の 指 定
０	１６進数表現のキャラクタコード８０～９Ｆ、Ｅ０～ＦＦを半角文字（1バイトコード文字）のコードとします。（＝ＫＭＯＤＥ０↵）
1	キャラクタコード８０～９Ｆ、Ｅ０～ＦＦを全角文字（2バイトコード文字）のコード（シフトＪＩＳコード）の第1バイトとします。（＝ＫＭＯＤＥ1↵）

d の値	使 用 モ ニ タ ー
０	専用ディスプレイテレビの標準／高解像度切換えスイッチの状態に従います。
1	標準ディスプレイの選択
2	高解像度ディスプレイの選択

m の値	設 定 さ れ る 画 面
０	テキスト画面２５行／グラフィック画面２００ライン
1	テキスト画面１２行／グラフィック画面１９２ライン
2	テキスト画面２０行／グラフィック画面なし
3	テキスト画面１０行／グラフィック画面なし
4	テキスト画面２５行／グラフィック画面４００ライン
5	テキスト画面１２行／グラフィック画面３８４ライン

〔例〕 ＫＭＯＤＥ1、標準ディスプレイ、テキスト画面１０行に設定します。

＊＃１１３↵ （パラメータ間にスペースを入れてはいけません。）

W (device Write)

働き： メインメモリの内容を指定デバイスの指定レコードに書き込みます。

文法： ＊WXd：nnnn rr aaaa

X： デバイス名。

M……………………グラフィック・メモリ（d＝０または1）

E……………………外部メモリ（d＝０～９）

D……………………３または５インチ版ディスク（d＝０～３）

F……………………８インチ版ディスク（d＝０～３）

H……………………５インチ１０Ｍハードディスク（d＝０～３）

d： ドライブ番号。(デバイス名Xによって範囲が異なります。)

nnnn： 書き込む先頭のレコード番号。最大4けたの16進数。

rr： 書き込む**レコード長**[1]。1～FFの16進数。

aaaa： メモリの先頭アドレス。0～FFFFの16進数。

1） レコード長 1単位が1セクタ（標準256バイト）。

説明： Xで指定されたデバイスのd番ドライブのレコード番号nnnnに、メインメモリ内のアドレス&Haaaaから始まるレコード長rrのデータを書き込みます。

デバイスXにDかFでディスクを指定するときは、Mコマンドを実行して、ディスクのタイプを示すデータをメモリのワーク・エリアに書き込んでおく必要があります。

次に、使用するディスクのドライブ番号と書き込むアドレスの関係とフロッピー・ディスクと書き込むデータの関係を示します。

使用するドライブ番号		ディスクのタイプを表わすデータを書き込むアドレス
3または5インチ版	ドライブ0	FAB4
	1	FAB5
	2	FAB6
	3	FAB7
8インチ版	ドライブ0	FAB8のビット1、0
	1	FAB8のビット3、2
	2	FAB8のビット5、4
	3	FAB8のビット7、6

フロッピーディスクのタイプ	書き込むデータ
3または5インチ版 2D（320K）	0
5インチ版 2DD（640K）	1
5インチ版 2HD（1M）	2
5インチ版 *2HD（1M、標準フォーマット）	3
8インチ版 2D—256（1M）	00
8インチ版 *2D—256（1M，標準フォーマット）	01
8インチ版 *1S—128（240K）	10

（8インチディスクのデータは2ビット毎のビットパターン1バイトから成っています。）

（例）　3または5インチのドライブ0で2DDタイプのフロッピーディスクを使用する場合

＊MFAB4↵

：FAB4＝01↵

：FAB5＝00 SHIFT ＋ BREAK

＊WD0：1　08　C000↵………3または5インチ版ディスクドライブ0のフロッピーディスク上のレコード番号1以降に、メインメモリ内のアドレス&HC000から8レコード分のデータを書き込みます。

Y (device read)

働き：　指定デバイスの指定レコードから指定のメインメモリアドレスにデータを読み込みます。

文法：　＊YXd：nnnn rr aaaa

X：　デバイス名。

M………グラフィック・メモリ（d＝0または1）

E………外部メモリ（d＝0～9）

D………3または5インチ版ディスク（d＝0～3）

F………8インチ版ディスク（d＝0～3）

H………5インチ10Mハードディスク（d＝0～3）

d：　ドライブ番号。（デバイス名Xによって範囲が異なる）

nnnn：　読み込む先頭のレコード番号。最大4けたの16進数。

rr：　読み込む**レコード長**[1]。1～FFの16進数。

aaaa：　メモリの先頭アドレス。0～FFFFの16進数。

1）レコード長　1単位が1セクタ（標準256バイト）。

説明：　Xで指定されたデバイスのd番ドライブのレコード番号nnnnから、メインメモリ内のアドレス&Haaaaから始まるエリアに、レコード長rrのデータを読み込みます。

デバイスXにDかFでディスクを指定するときは、Mコマンドを実行して、ディスクのタイプを示すデータをメモリのワーク・エリアに書き込んでおく必要があります。書き込むデータについてはWコマンドを参照してください。

12章
ディスクの使い方

本機では、5インチミニフロッピーディスク（3インチコンパクトフロッピーディスク）のほかに8インチフロッピーディスク、5インチハードディスクをＢＡＳＩＣでサポートしています。ここではこれらのフロッピーディスク装置を入力出力装置として使う場合に、必要な事柄を説明しています。

なお、プログラムやデータを保存または再生するときは「8章　プログラム、データファイルの保存と再生」をご覧ください。

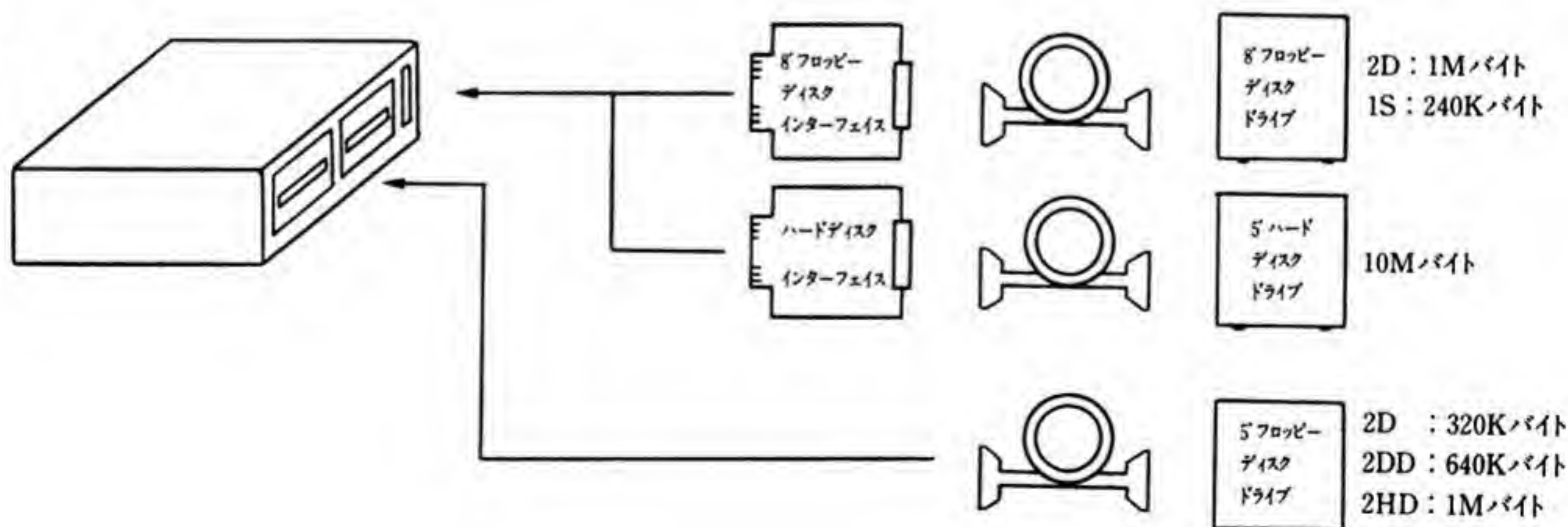
1　ディスクの使用準備

新しいフロッピーディスクを使用される場合、あらかじめフロッピーディスクのフォーマッティングを行なっておく必要があります。

フォーマッティングとは新らしいフロッピーディスクを使用できるようにすることで、ＤＩＳＫ ＢＡＳＩＣのユーティリティプログラムのためのプログラムが用意されていますので、フロッピーディスクのフォーマッティングの手順に従って行なってください。

（別冊の『アプリケーションソフトの説明書』の「ディスクユーティリティ」参照）

2　ドライブ接続構成

本機は下記構図の様に各々3インチまたは5インチフロッピーディスクドライブ、（容量：320Kバイト、640Kバイト、1Mバイト）、8インチフロッピーディスクドライブ（容量：1Mバイト、240Kバイト）および5インチハードディスクドライブ（容量10Mバイト）を接続可能です。

ドライブ接続構成図

本機は5インチフロッピーディスクドライブ及びインターフェイスを内蔵しているので3インチまたは5インチフロッピーディスクドライブは直接増設可能であり、8インチフロッピーディスクドライブ、ハードディスクドライブは専用インターフェイスボードを使用して外部接続ができます。
〈ドライブ接続台数〉

接続ドライブ		台数	備考
3インチまたは5インチフロッピーディスクドライブ	内蔵	2	最大接続台数はトータルで4台
	内部増設	—	
	外部増設	2	
8インチフロッピーディスクドライブ		4	専用インターフェイス使用
5インチハードディスクドライブ		4	専用インターフェイス使用

(サポートドライブを最大接続した時の容量は48Mバイトです。)
本機には外部5インチフロッピーディスクインターフェイスは接続しないでください。

3 システムプログラム起動方法

本機では、3インチまたは5インチフロッピーディスクドライブをトータル4ドライブ、8インチフロッピーディスクドライブを4ドライブ(専用インターフェイス使用)およびハードディスクドライブを4ドライブ(専用インターフェイス使用)が一度に接続可能であり、8種類のディスクタイプ(次表参照)をサポートしているのでシステムプログラムを起動する場合あらかじめシステムディスクの入っているドライブナンバーとディスクタイプを設定する必要があります。その方法として

(1)初期モードスイッチ(ディップスイッチ)により、システムディスクのタイプ設定
(2)キー入力により、システムディスクのタイプ設定

の2種類が可能です。

(1)初期モードスイッチによるシステムディスクのタイプ設定

この方法は、初期モードスイッチにより起動するシステムディスクナンバーを設定し、システム電源を投入することによってサポートドライブ(3インチまたは5インチおよび8インチフロッピーディスクドライブ、ハードディスクドライブ)のドライブ0から自動的にシステムプログラムが起動できます。

*出荷時ディップスイッチは全てON(ディスクタイプナンバー0)に設定しています。

ディスクタイプナンバーとディスクの種類

ディップスイッチ			ディスクタイプナンバー	内容	記録方式	記録内容	ディスク名
SW1	SW2	SW3					
ON	ON	ON	0	2D X1 フォーマット	両面 倍密度記録	３２０K	3チンチ 5インチFD
OFF	ON	ON	1	2DD X1 フォーマット	両面倍トラック 倍密度記録	６４０K	
ON	OFF	ON	2	2HD X1 フォーマット	両面高密度 倍密度記録	１M	
OFF	OFF	ON	3	＊2HD 8インチ 標準フォーマット	両面高密度 倍密度記録	１M	
ON	ON	OFF	4	2D X1 フォーマット	両面 倍密度記録	１M	8インチFD
OFF	ON	OFF	5	＊2D 8インチ 標準フォーマット	両面 倍密度記録	１M	
ON	OFF	OFF	6	1S 8インチ 標準フォーマット	片面 単密度記録	２４０K	
OFF	OFF	OFF	7	X1 フォーマット	4ヘッド 倍密度記録	１０M	ハードディスク

＊但しヘッド0、シリング0のみ１セクター＝１２８バイトの単密度記録

またシステムプログラムを起動したサポートドライブ0に対するディスクタイプも同時に、システムのディスクタイプ設定レジスタに書き込まれ、以後のプログラム上で、ドライブのデバイス名を省略してアクセスしようとした場合は、このドライブナンバーが指定されます。

一般にこの様にシステムプログラムを起動したドライブを、**カレントドライブ**と呼んでいます。

一方、システムプログラムが起動できなかった時は

```
      Make your device ready
Press selected key to start driving:
             F:Floppy
             R:ROM
             C:CMT
             T:Timer
```

という表示となるので初期モードスイッチで設定したディスクタイプのドライブ状態の再確認後、再び電源を入れてシステムプログラムを起動するか、キーボード入力での操作で、システムプログラムを起動してください。

(2)キー入力でシステムプログラムを起動する場合

初期モードスイッチの設定に関係なくキー入力によって、ドライブの選択を行ない任意のディスクタイプ（ナンバー0から7）のディスクから、システムプログラムが起動できます。またそのドライブナンバーがカレントドライブに設定され、デバイス名を省略してアクセスした時はカレントドライブ（システムプログラムを起動したドライブ）が指定されます。

操作方法

本機の電源をＯＮすると

```
     Make your device ready
 Press selected key to start driving:
           F:Floppy
           R:ROM                              ………(1)
           C:CMT
           T:Timer
```

という表示が画面に出ますので、キーボードから F キーを入力します。

```
 Drive No ?   (0-3)                           ………(2)
```

という表示となりますので、次にシステムプログラムを起動しようとするドライブナンバー（0～3）を指定するために、数字キーの 0 ～ 3 のうち指定のナンバーをキー入力すると、

```
 Type of DISK ?   (0-7)

 5"FD  320kbytes    2D          ……0
  or
 3"FD  640kbytes    2DD         ……1
       1M  bytes    2HD         ……2
       1M  bytes   *2HD         ……3
                                              ………(3)
 8"FD  1M  bytes    2D-256      ……4
       1M  bytes   *2D-256      ……5
       240kbytes   *1S-128      ……6

 5"HD  10M bytes                ……7
```

とサポートしている全てのディスクタイプが表示されますので、起動したいディスクタイプに対応しているナンバーを指定するために、数字キーの 0 ～ 7 キーを使って入力すると任意のドライブナンバーから任意のディスクタイプでシステムプログラムが起動されます。表示画面は

```
     IPL is lookig for a program from FD NO   ………(4)
```

（ＮＯは0～3のうちどれか1つ）

となり、設定したディスクタイプで読み出し可能な時は

```
IPL is loading BASIC CZ8FB02                ………(5)
```

表示となり、ＢＡＳＩＣの読み出しを開始します。

一方設定したドライブナンバーから、設定したディスクタイプで読み出し不可能だった時は(1)の表示になるので、入出力装置の状態を確認の上、再操作してください。また、(2)または(3)の画面の時 SHIFT ＋ BREAK キーを押しますと、(1)の表示画面に戻すことができます。入力ミスをした場合などに利用してください。

キーボードから設定して、システムプログラムを起動した場合、起動したドライブに対するディスクタイプのナンバーは自動的にシステムのディスクタイプ設定レジスタに書き込まれます。他のサポートドライブに対しては初期設定(３インチまたは５インチドライブの場合ディスクタイプ０、８インチドライブの場合はディスクタイプ４）されているので、各ドライブに対してディスクタイプを設定する必要があります。

4 接続ドライブのディスクタイプ設定

本機では、計１２ドライブと接続が可能です。

		デバイス名
・３インチまたは５インチ(２Ｄ、２ＤＤ、２ＨＤ)	………４ドライブ ；	０：～３：
・８インチ（２Ｄ、１Ｓ)	………４ドライブ ；	Ｆ０：～Ｆ３：
・ハードディスク	………４ドライブ ；	ＨＤ０：～ＨＤ３：

３インチまたは５インチおよび８インチのフロッピーディスクドライブにどのタイプのディスクが挿入されているか、本機で認識する必要があります。本機で認識しているディスクタイプと違うディスクを挿入すると、書き込み／読み出しができない場合が生じます。すなわちディスクタイプによってヘッド数、シリング数、セクタ数が異なり、また、ＦＭ記録／ＭＦＭ記録、３インチまたは５インチ（２Ｄ、２ＤＤ）／５インチ（２ＨＤ)、８インチ（２Ｄ、１Ｓ）の切換えを各フロッピーインターフェイスに指示する必要があるためです。

ディスクタイプを正確に認識することによって、自動的にプログラムの書き込み／読み出しの管理ができます。

ディスクタイプの設定方法として

(1) ユーティリティプログラムの実行
(2) ＤＥＶＩＣＥ命令
(3) ワークレジスタの書き換え

の３方法があります。

(1)は、ユーティリティプログラムを走らせ各デバイス名に対応するディスク・タイプＮo.をキーボードから入力すると自動的に設定できます。

(2)は、ＢＡＳＩＣの ＤＥＶＩＣＥ命令で

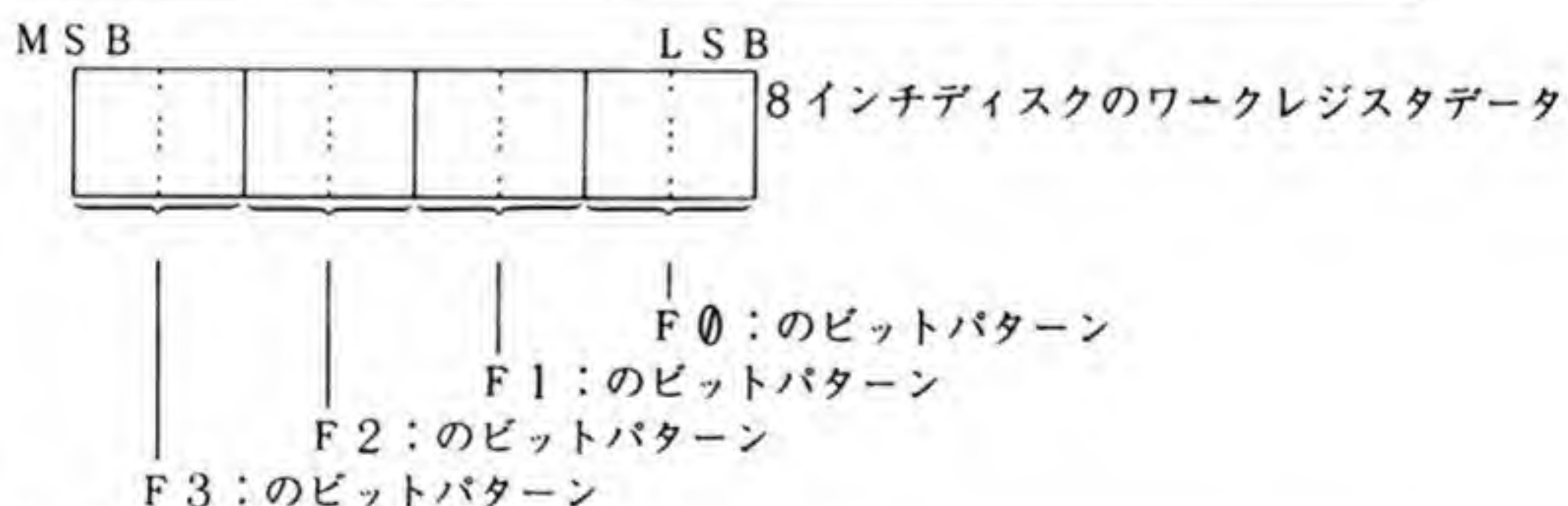
ＤＥＶＩＣＥ "デバイス名：ディスクタイプＮo."↵

によって設定できます。(次表参照)

	デバイス名	ディスクタイプNo
3インチ または 5インチ FD	Ø： ≀ 3：	Ø… 2D 1… 2DD 2… 2HD 3… 2HD
8インチ FD	FØ： ≀ F3：	Ø… 2D 1… 2D 2… 1S

(3)はモニタコマンドまたはBASICで、ワークレジスタを変更する方法です。

ディスクタイプのワークレジスタは、各デバイス名について5インチディスクは1バイト、8インチディスクは2ビット毎のビットパターン1バイトから成っています。

	デバイス名	ワークレジスタアドレス	データ
5インチ FD	Ø：	FAB4	Ø：2D X1フォーマット
	1：	FAB5	1：2DD X1フォーマット
	2：	FAB6	2：2HD X1フォーマット
	3：	FAB7	3：2HD 8インチ標準フォーマット
8インチ FD	FØ：	FAB8	ØØ：2D X1フォーマット
	F1：		Ø1：2D 8インチ標準フォーマット
	F2：		1Ø：1S 8インチ標準フォーマット
	F3：		(ビットパターン)

例えば　接続ディスクが

3インチまたは5インチ FD	Ø：	2D	8インチ FD	FØ：	2D X1フォーマット
	1：	2D		F1：	2D X1フォーマット
	2：	2DD		F2：	2D 8インチ標準フォーマット
	3：	2HD X1 フォーマット		F3：	1S 8インチ標準フォーマット

の場合

ディスクタイプのワークレジスタは

FAB4 ; 00
FAB5 ; 00
FAB6 ; 01
FAB7 ; 02
FAB8 ; 90 となります。

ディスクタイプのデフォルト（初期モードスイッチの設定の場合）

(1)3インチまたは5インチディスクでBOOTした時

デバイス名	デフォルトタイプ
0：	BOOTしたディスクタイプ
1：～3：	2D
F0：～F3：	2D X1フォーマット

(2) 8インチディスクでBOOTした時

デバイス名	デフォルトタイプ
0：～3：	2D
F0：	BOOTしたディスクタイプ
F1：～F3：	2D X1フォーマット

(3) ハードディスクでBOOTした時

デバイス名	デフォルトタイプ
0：～3：	2D
F0：～F3：	2D X1フォーマット

また、キー入力で、BOOTした時は、BOOTしたドライブのデバイス名が、そのディスクタイプに設定され、他の3インチまたは5インチ、8インチフロッピーディスクドライブは

3インチまたは5インチFD… 2D
8インチFD… 2D X1フォーマット

に設定されます。

5 ディスクのフォーマット構成

ディスクにデータの書き込み読み出し動作を行なう場合、ディスク上のアドレス情報（ヘッドNo.、シリンダーNo.、セクターNo等）を基にして、データの格納場所を検索し実行します。こ

のため、ディスク上にこのアドレスを割り付ける必要があります。この動作をフォーマット（1次フォーマット）といいます。

3インチまたは5インチＦＤ２Ｄ、２ＤＤの市販ディスクはフォーマットされていないディスクなので、必ずフォーマットする必要があります。一方、5インチＦＤ２ＨＤおよび8インチＦＤ２Ｄ、１Ｓは8インチＦＤ標準フォーマットでフォーマットされているので、必ずしもフォーマットする必要がありませんが全レコード２５６バイト／セクター、倍密度記録のＸ１フォーマット注1）に、フォーマットして使用してもかまいません。
サポートしているディスクタイプおよびフォーマットを下記に示します。

ディスク ＼ 項目		ヘッドNo	シリンダNo	セクターNo	セクターデータ（バイト）	内　容
3インチまたは5インチFD	2D	0,1	0～39	1～16	256	X1 フォーマット
	2DD	0,1	0～79	1～16	256	X1 フォーマット
	2HD	0,1	0～76	1～26	256	X1 フォーマット
		0,1	0～76	1～26	256 注2） ヘッド0 シリンダー0 128	8インチFD 標準フォーマット
8インチFD	2D	0,1	0～76	1～26	256	X1 フォーマット
		0,1	0～76	1～26	256 注2） ヘッド0 シリンダー0 128	8インチFD 標準フォーマット
	1S	0	0～76	1～26	注3） 128	8インチFD 標準フォーマット
ハードディスク		0～3	0～305	1～33	256	X1 フォーマット

注1）Ｘ１フォーマットは、全てのシリンダーに対して
２５６バイト／セクターの倍密度記録
注2）８インチＦＤ標準フォーマットは
ヘッドＮｏ．0、シリンダーＮｏ．0のセクターＮｏ．１～２６は、
１２８バイト／セクターの単密度記録
その他の領域は
２５６バイト／セクターの倍密度記録
注3）全てのシリンダに対して
１２８バイト／セクターの単密度記録

また、システムでディスク管理を行なうためにはディスク上にある特別な領域、ＦＡＴ領域とディレクトリ領域を設け、システムに合うようにＦＡＴ領域およびディレクトリ領域に初期データを書き込む必要があります。これをシステムフォーマット（2次フォーマット）と呼んでいます。本機ではディスク管理する場合は、全てのディスクタイプとも必ずシステムフォーマットする必要があります。

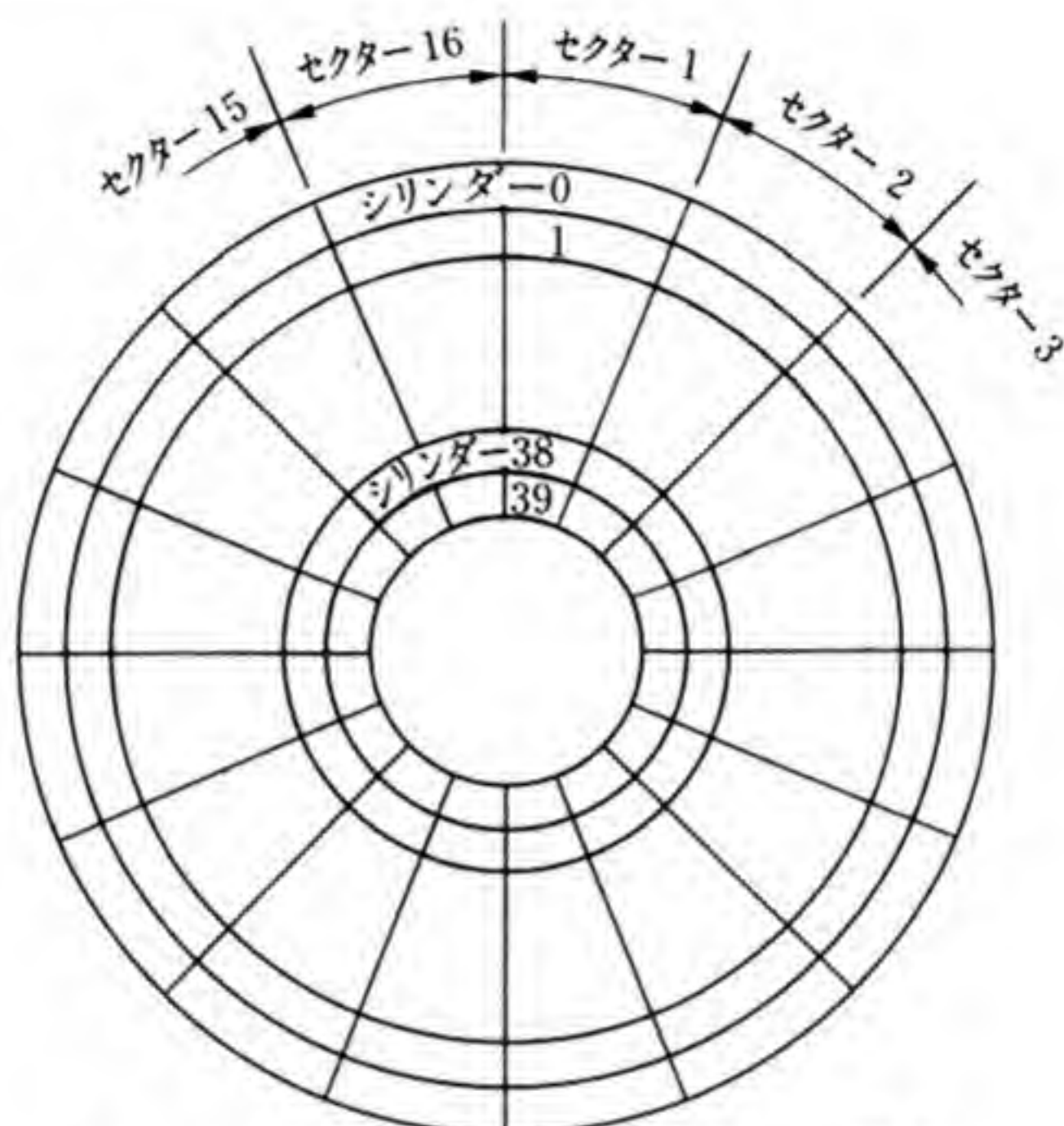
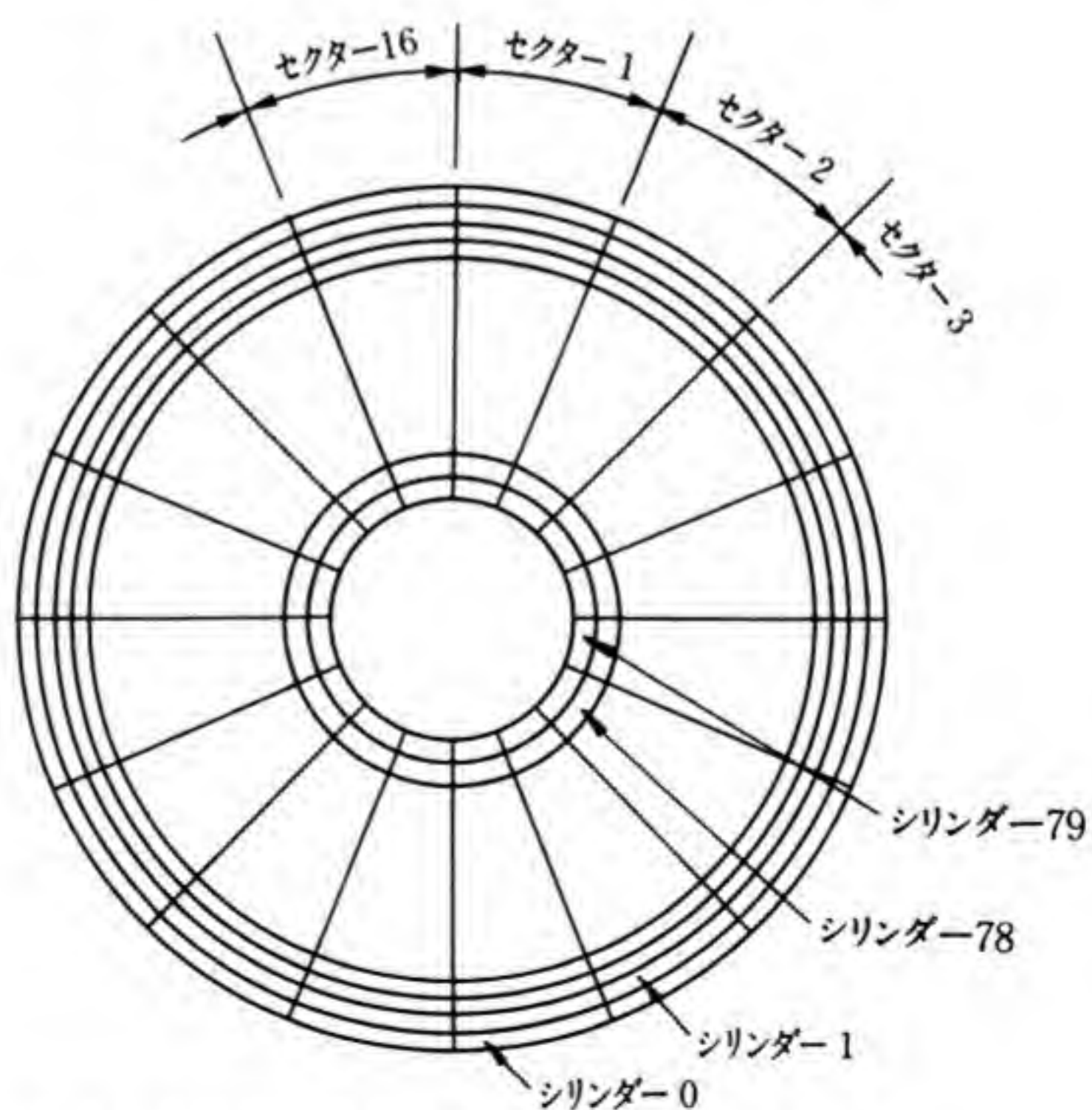
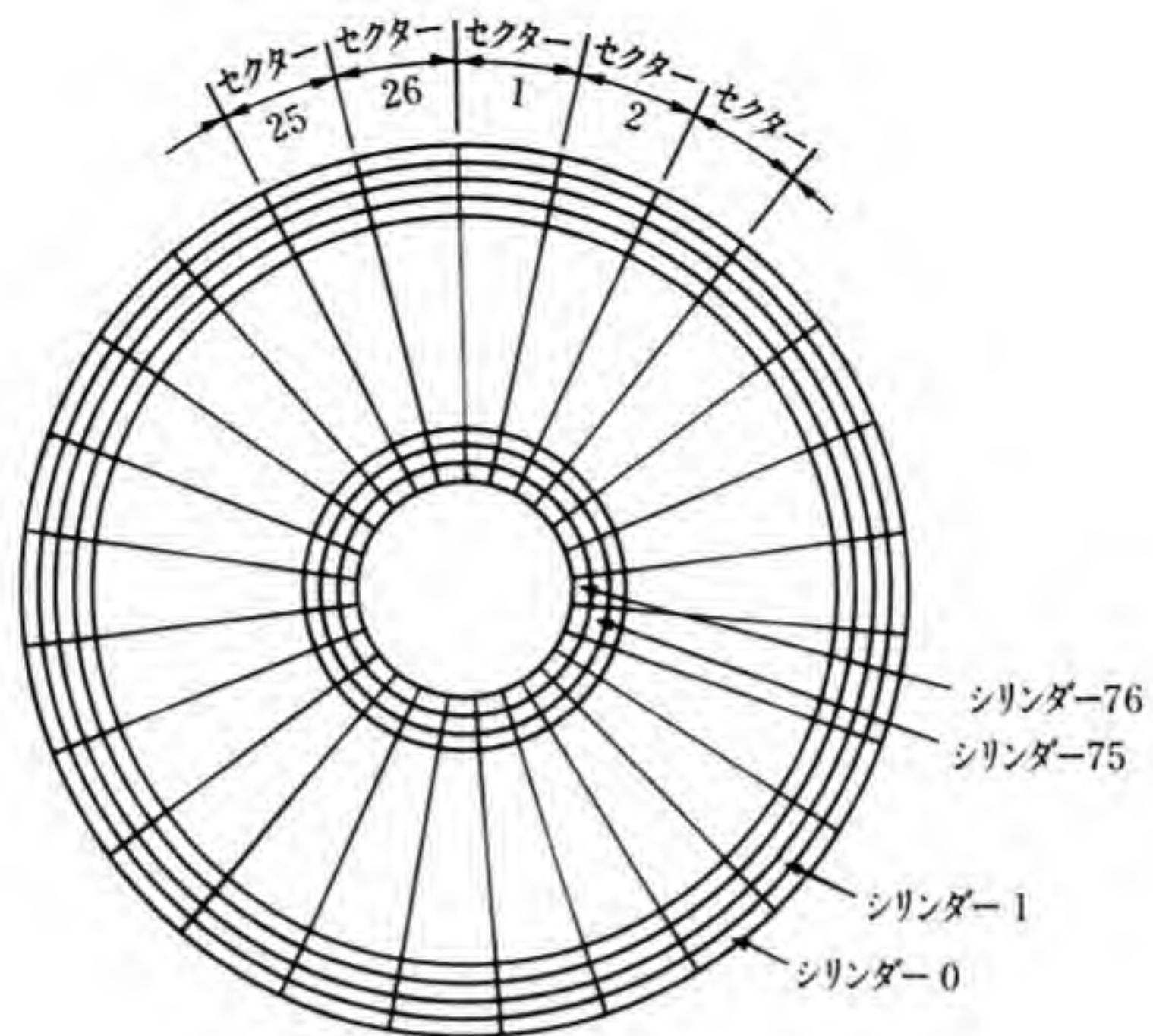
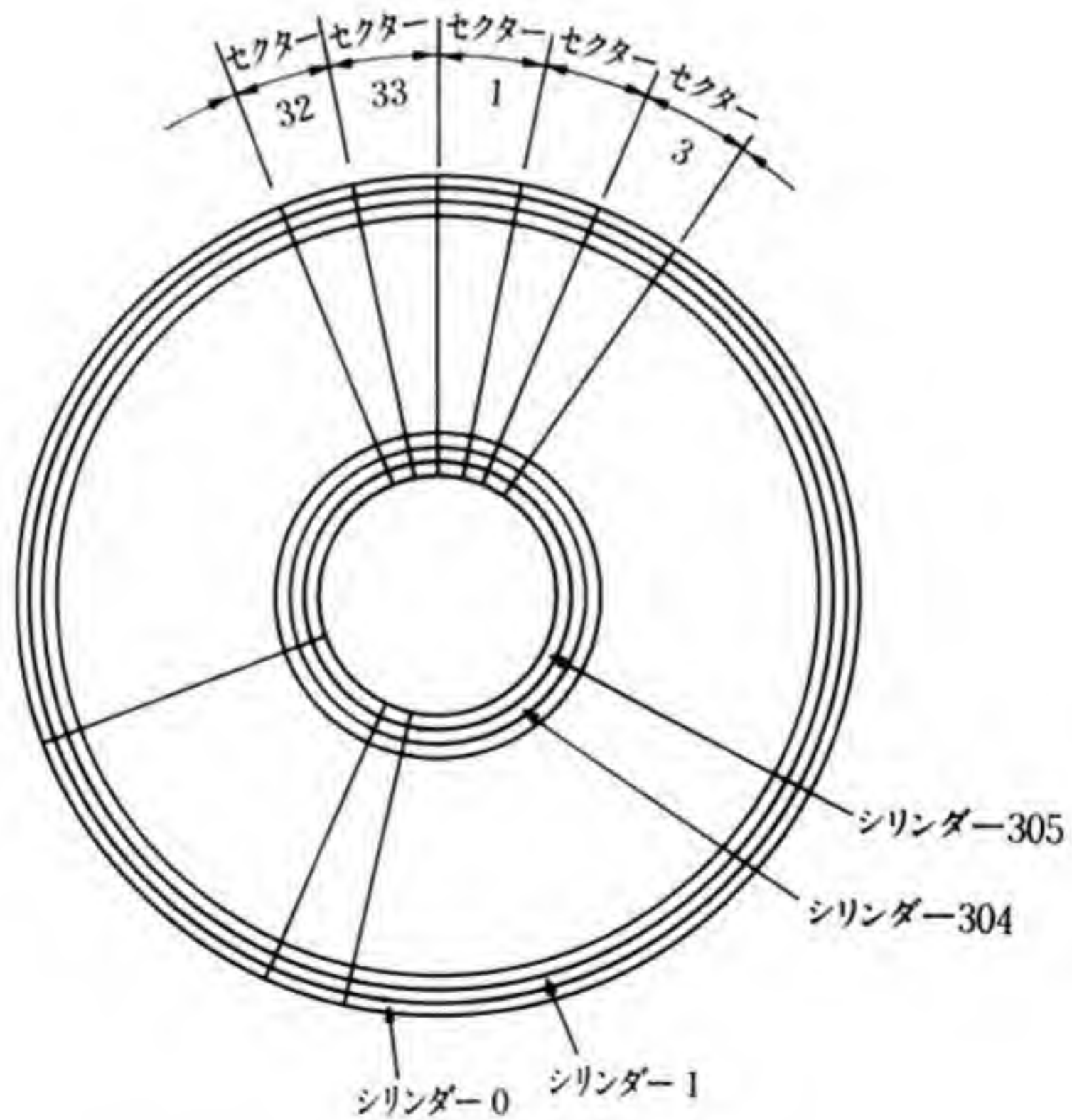
実際にディスクをフォーマットする場合は

ユーティリティプログラム中のフォーマットプログラムで実行してください。

詳しくは『アプリケーションソフトの説明書』の「ディスクユーティリティ」をお読みください。

このフォーマットプログラムでは、１次フォーマットと２次フォーマットを両方行ないますが、５インチＦＤ・２ＨＤおよび８インチＦＤ・２Ｄ、１Ｓの市販ディスクは２次フォーマット（システムフォーマット）のみで使用可能です。

このシステムフォーマットを行なう場合は、ＩＮＩＴ"デバイス名："を実行してください。

（『ＢＡＳＩＣ　リファレンス　マニュアル』2.4.1 ＩＮＩＴ参照）

５インチＦＤ　２Ｄタイプのディスク構造

５インチＦＤ　２ＤＤタイプのディスク構造

5インチＦＤ　２ＨＤ
8インチＦＤ　２Ｄ　１Ｓ　タイプのディスク構造

５インチハードディスクのディスク構造

6　ディスクファイル管理方法

３インチまたは5インチ、８インチフロッピーディスクおよび、ハードディスクの全てのタイプのディスクへの書き込み／読み出しは、ヘッド番号、シリンダー番号、セクター番号から連続した論理アドレスに変換し、その最小単位のレコード（１２８バイト／２５６バイト）ごとに、可能ですが実際のファイル管理での最小単位は、１６レコード＝１クラスタ（４Ｋバイト）で行なっています。

ディスク上におけるファイルの記録状態や使用状態の管理は、ＦＡＴ（Ｆｉｌｅ　Ａｌｌｏｃａｔｉｏｎ　Ｔａｂｌｅ）のクラスタ番号とディレクトリデータで行なっています。各ディスクタイプの記録容量によって　ＦＡＴ、ディレクトリ、データ領域は下記の構成となっています。

（単位＝レコード）

３インチまたは５インチフロッピーディスク				8インチフロッピーディスク	ハード
ディスク 項目	2D	2DD	2HD	2D	ディスク
システム領域	Ø	Ø	Ø	Ø	Ø
ＦＡＴ領域	14	14 〜 15	28 〜 29	28 〜 29	8 〜 27
ディレクトリ領域	16 〜 31	16 〜 31	32 〜 47	32 〜 47	48 〜 63
データ領域	32 〜 1279	32 〜 2559	48 〜 4ØØ3	48 〜 4ØØ3	64 〜 4Ø127
代替領域	—	—	—	—	4Ø128 〜 4Ø391

ハードディスクの使用終了時に、ヘッドをデータ領域以外に移動することにより、ディスクデータの破壊からの保護を行なっています。

８インチＦＤの片面単密度記録は　１レコード単位の書き込み／読み出しは可能であるがＢＡＳＩＣでのファイル管理はサポートしていません。

(1)ディレクトリ

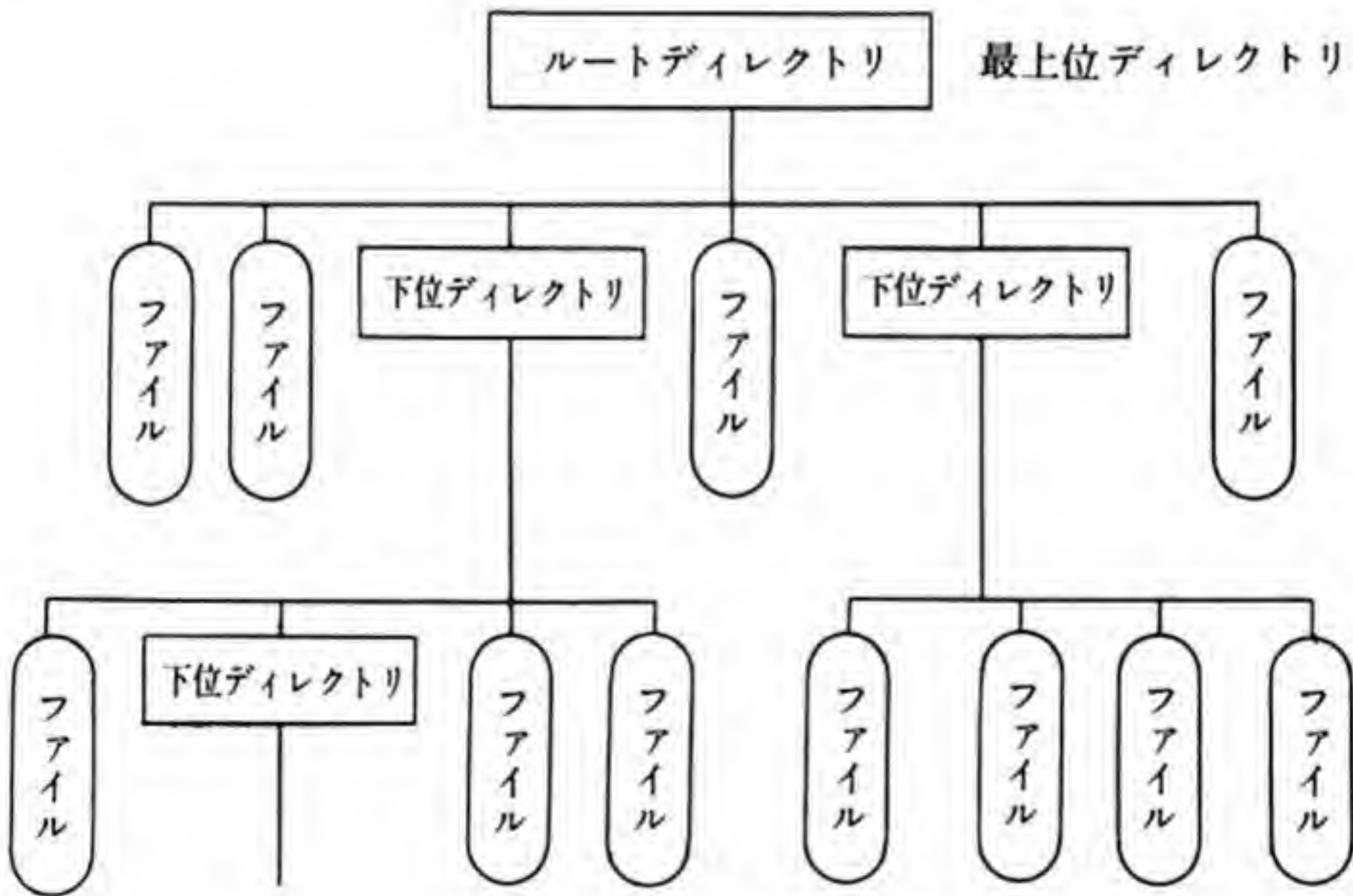
ディレクトリはファイル管理のためのファイル名、格納先頭クラスタ番号、データ数、日付等を表わし、１ファイル３２バイトで表現しています。

各タイプのディスクとも階層ディレクトリ形式なので、データ領域内に下位ディレクトリが設定され、各下位ディレクトリはディスクタイプに関係なく、１クラスタ分（４Ｋバイト：１２８ファ

イル数）の領域を確保できます。最上位ディレクトリ（ルートディレクトリ）で管理できるファイル数はディスクタイプで異なり次表に示すとおりです。

ディスクタイプで
管理可能な最大ファイル数

項目 ディスク		ルートディレクトリ 領域　レコード数	ファイル数
3インチ または 5インチ FD	2D	16	78
	2DD	16	158
	2HD	16	247
8インチFD	2D	16	247
ハード ディスク		16	2504

ディレクトリの構成は下図の様なツリー構造になります。

ディレクトリのファイルに対する構成を次に示します。（1ファイル＝32バイト）

1ファイル名に対するディレクトリの構成

	内　　容
1バイト目	種類を表わす。　ØØはKILLされたファイルまたは未使用領域。FFは使用ディレクトリテーブルの終り。 bit Øが1 ……Bin ファイル（機械語で書かれたファイル） bit 1が1 ……Bas ファイル（BASICテキストで書かれたファイル） bit 2が1 ……Asc ファイル（ASCIIセーブされたファイル） bit 4が1 ……FILESで表示しない：Ø…表示する bit 5が1 …リードアフターライトON：Ø…OFF bit 6が1 …書き込み禁止ファイル：Ø…書き込みOK bit 7が1 …下位ディレクトリ bit 3は予備
2バイト目～14バイト目	ファイル名（13文字）
15バイト目～17バイト目	ユーザー指定　エクステンションエリア　（3文字）
18バイト目	パスワードのバック（無指定なら2Ø（16）の値）
19・2Øバイト目	ファイルのバイト数（BasおよびObjのみ有効）
21・22バイト目	ファイルのメインメモリー先頭アドレス（Objのみ有効）
23・24バイト目	ファイルのメインメモリー実行アドレス（Objのみ有効）
25バイト目～29バイト目	作成された年、月、曜日、日、時、分が書き込まれています。 例：　'84年12月Ø1日土曜日、16時36分 先頭から、年年　月　曜　日日　時時　分分 84　C　6　Ø1　16　36
3Øバイト目～32バイト目	ファイル先頭　クラスタ値 3Øバイト目　HIGHバイト 31バイト目　LOWバイト 32バイト目　MIDDLEバイト

(2)　FAT

ＦＡＴはディスクで管理されるファイルのチェーン状態を表わし、２バイトのクラスタ番号で表現しています。ＦＡＴの必要容量（バイト数）は（管理可能な最大ファイル数）×２バイトです。

ＦＡＴ構成としては

アドレス		
0 ～ 7F	0～7F 80～8F	下位バイト
80 ～ FF	0～13	上位バイト

図の様にＦＡＴ領域の１レコードを１２８バイトに分割し、各々をクラスタの下位、上位バイトに指定しています。

各ディスクタイプによってディスク管理用の予約バイトとして　ＦＡＴ領域の先頭から

3インチまたは5インチFD 2D,2DD	2バイト
5インチFD 2HD 8インチFD 2D	3バイト
ハードディスク	4バイト

上表のように必要です。ＦＡＴデータとしては、次のようなものがあります。

項目＼ディスク	3インチまたは5インチFD			8インチFD	ハードディスク
	2D	2DD	2HD	2D	
ファイルがチェーンしているクラスタ	02～4F	02～7F 100～11F	03～7F 100～179	03～7F 100～179	04～7F 100～17F 1300～1353
ファイル終了クラスタ	80～8F	80～8F	80～8F	80～8F	80～8F
未使用クラスタ	0	0	0	0	0

13章

RS-232Cインターフェイスの使い方

本機はＲＳ－２３２Ｃインターフェイスを内蔵しており、ＲＳ－２３２Ｃインターフェイスを持っている機器との間でシリアルデータの送・受信ができます。

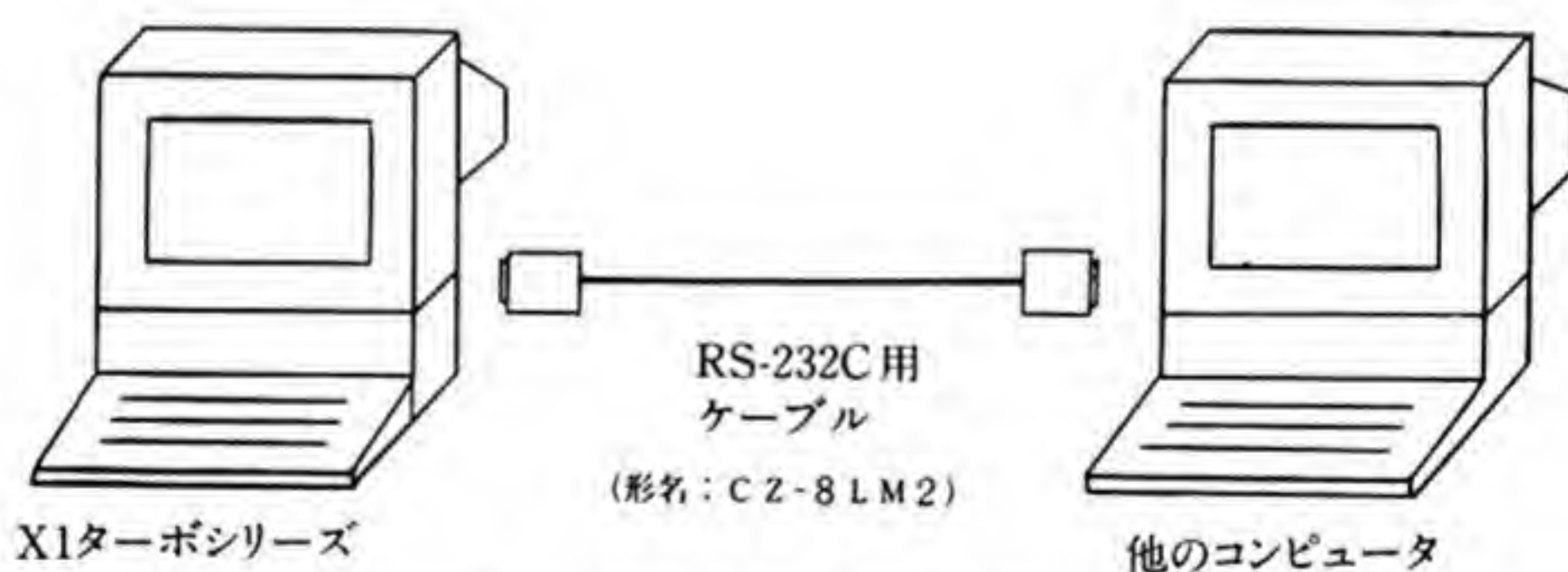
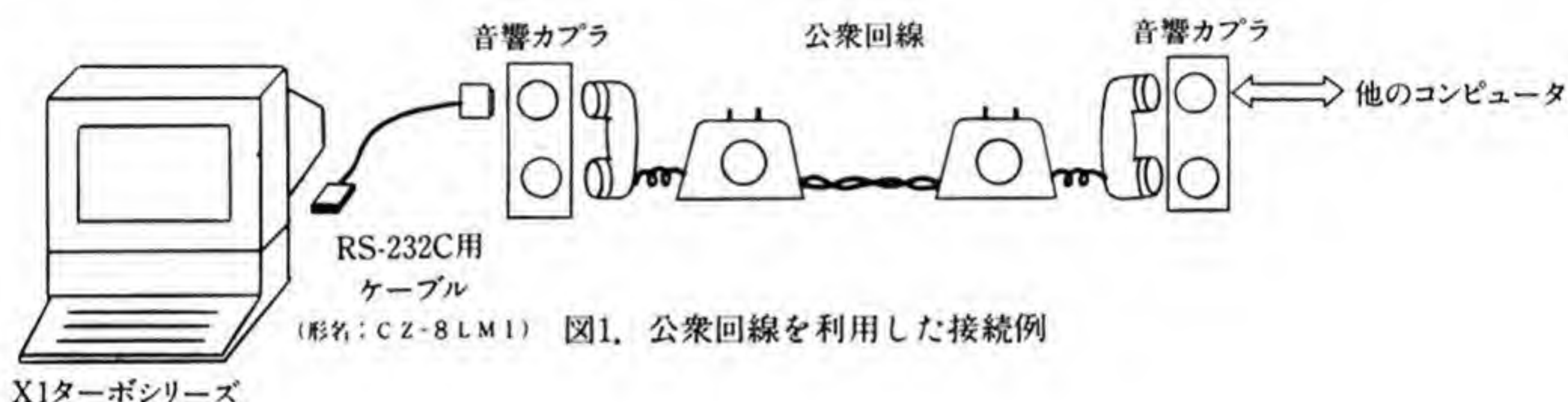
1　接続方法

(1) 接続例

ＲＳ－２３２Ｃインターフェイス機能を使用して、以下に示すようなシステムのもとで、ＲＳ－２３２Ｃインターフェイス機能を持つ他のコンピュータあるいは周辺機器との間でデータの送・受信ができます。

図1. 公衆回線を利用した接続例

図2. 他のコンピュータとの接続

13章

(2) ＲＳ-２３２Ｃインターフェイス信号端子表

ＲＳ－２３２Ｃインターフェイス用コネクタには２５ピンＤ―ｓｕｂコネクタのメス形を使用しており、そのピン配置と信号端子は次のとおりです。

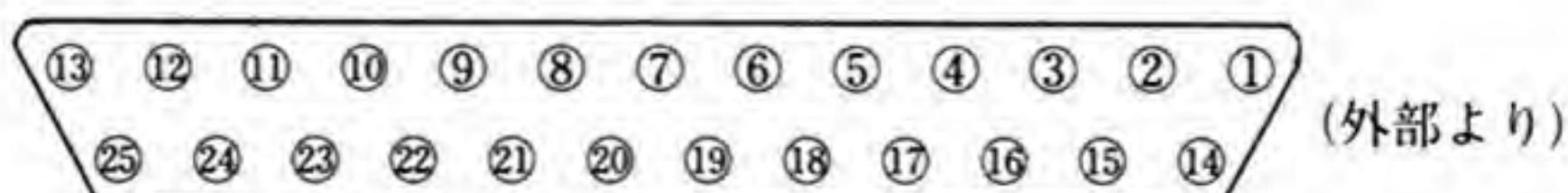



図３．ＲＳ-232Ｃインターフェイスコネクタピン配置

RS－232Cコネクタ端子表

端子番号	信　号　名	信号方向	機　　　能
1	FG（保安用アース）	←→	保安用アース
2	TxD（送信データ）	→	送信データ
3	RxD（受信データ）	←	受信データ
4	RTS（送信要求）	→	ONにより相手を送信可能とする
5	CTS（送信可）	←	ONの時送信可能と判断する
6	DSR（データセットレディ）	←	ONの時相手が送・受信の準備ができていると判断する
7	SG（信号用アース）	←→	信号用アース
8	CD（キャリア検出）	←	ONの時受信データ有効と判断する
9～14	NC		
15	ST2（送信信号エレメントタイミング）	←	同期式通信時の送信信号エレメントタイミング信号を入力する
16	NC		
17	RT（受信信号エレメントタイミング）	←	同期式通信時の受信信号エレメントタイミング信号を入力する
18,19	NC		
20	DTR（データターミナルレディ）	→	ONにより送・受信の準備ができたことを知らせる
21	NC		
22	CI（被呼表示）	←	ONにより相手から呼び出されていると判断する
23	NC		
24	ST1（送信信号エレメントタイミング）	→	同期式通信時の送信信号エレメントタイミングを出力する
25	NC		

→OUT
←IN

(3) 接続方法

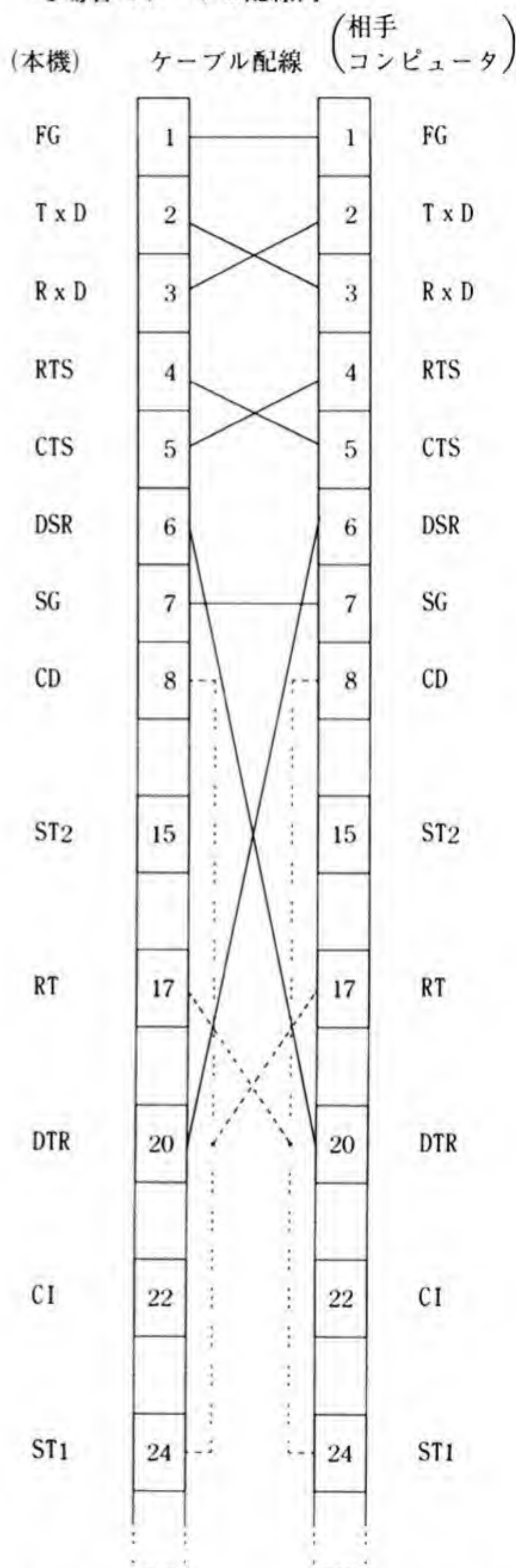
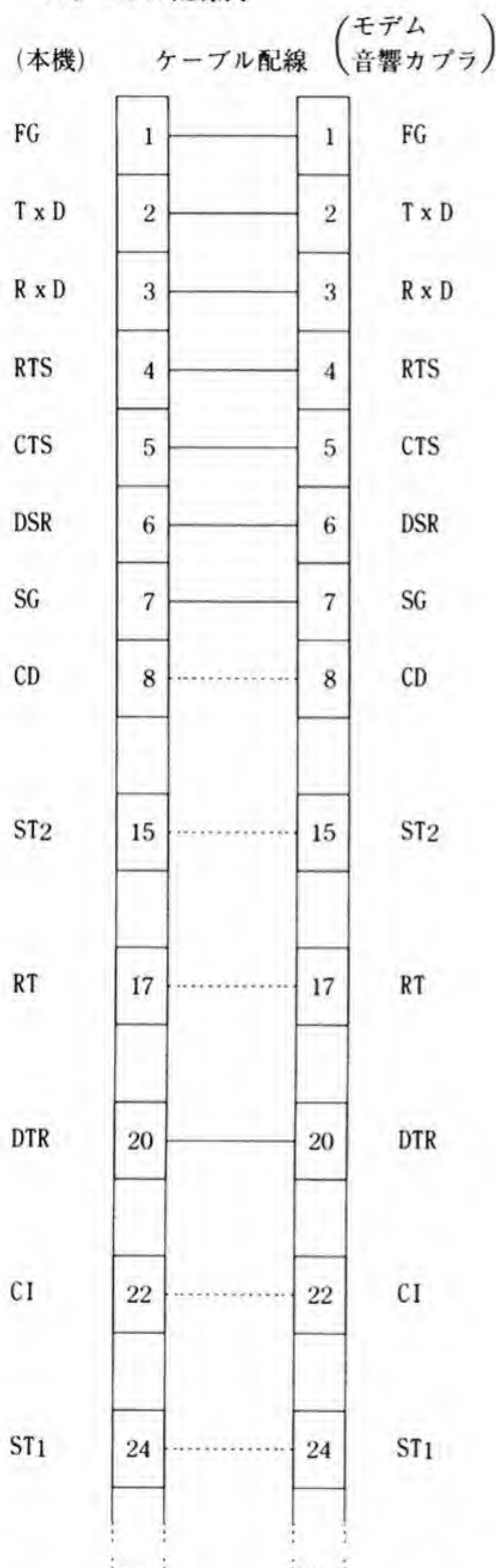
相手機器との接続には両端に25ピンD－subコネクタのオスを持つRS－232C用ケーブルを使用します。相手機器がモデム（音響カプラ含む）かコンピュータかによりコネクタ信号の方

向が異なります。相手機器によっては使用コネクタまたは信号配置が異なるものもあるため、相手機器の説明書を熟読の上接続してください。以下に一般的な接続例を示します。

図4　モデムや音響カプラと接続する場合のケーブル配線例

(本機)		ケーブル配線		(モデム 音響カプラ)
FG	1		1	FG
TxD	2		2	TxD
RxD	3		3	RxD
RTS	4		4	RTS
CTS	5		5	CTS
DSR	6		6	DSR
SG	7		7	SG
CD	8		8	CD
ST2	15		15	ST2
RT	17		17	RT
DTR	20		20	DTR
CI	22		22	CI
ST1	24		24	ST1

図5　他のコンピュータ（本機を含む）と接続する場合のケーブル配線例

(本機)		ケーブル配線		(相手 コンピュータ)
FG	1		1	FG
TxD	2		2	TxD
RxD	3		3	RxD
RTS	4		4	RTS
CTS	5		5	CTS
DSR	6		6	DSR
SG	7		7	SG
CD	8		8	CD
ST2	15		15	ST2
RT	17		17	RT
DTR	20		20	DTR
CI	22		22	CI
ST1	24		24	ST1

…で示す信号線はＢＡＳＩＣではサポートしておらず特に接続する必要はありません。外部同期を使用する場合など、ユーザーがマシン語レベルでソフトを作成して使用する時に必要に応じて接続します。

2　RS-232インターフェイスのデータ信号フォーマット

RS－232インターフェイスは、コンピュータからの8ビットパラレルデータをシリアルデータに変換してRS－232Cで規定された電圧レベル（±12V）に変換して送信したり、逆に受信したシリアルデータをパラレルデータに変換するものです(図6参照)。シリアルデータの形式は同期式、非同期式通信方法によって異なります。**本機のBASICでは非同期通信方式をサポートしています**。この場合のデータ信号フォーマットを図7に示します。機器間でデータの送・受信を行なう場合にはこのデータ信号フォーマットおよびボーレート（信号の伝送速度）を一致させる必要があります。設定方法については次項のBASIC上でのRS－232Cインターフェイス取扱方法で説明します。

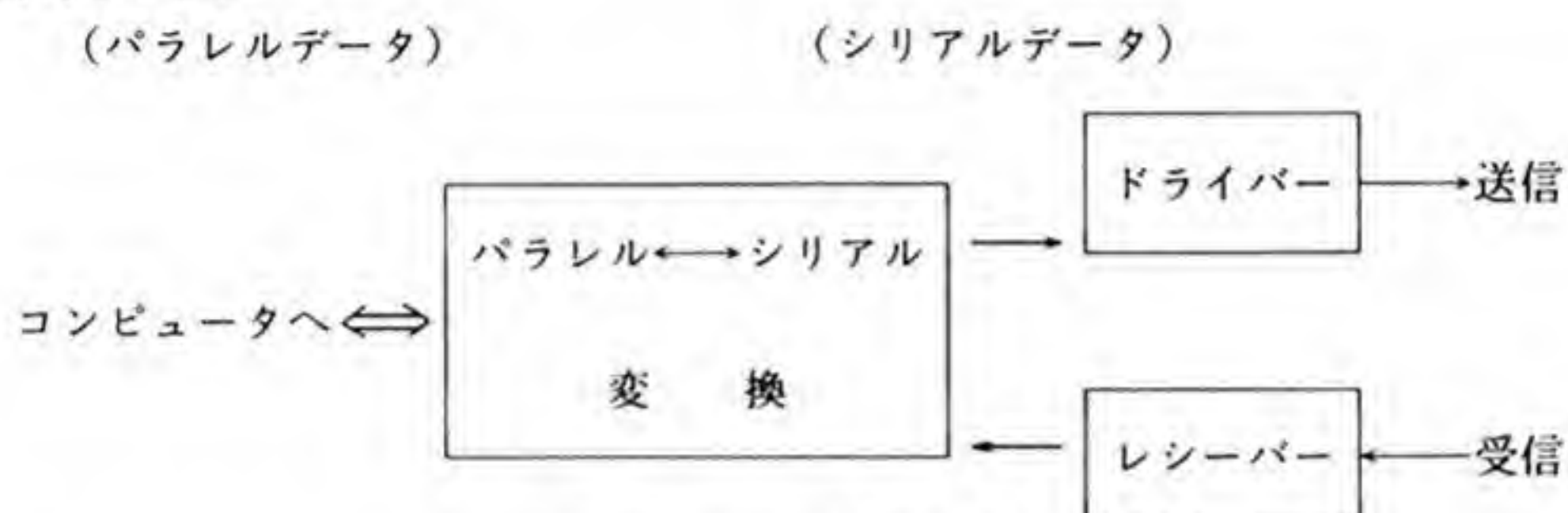



図6　RS－232C概念図

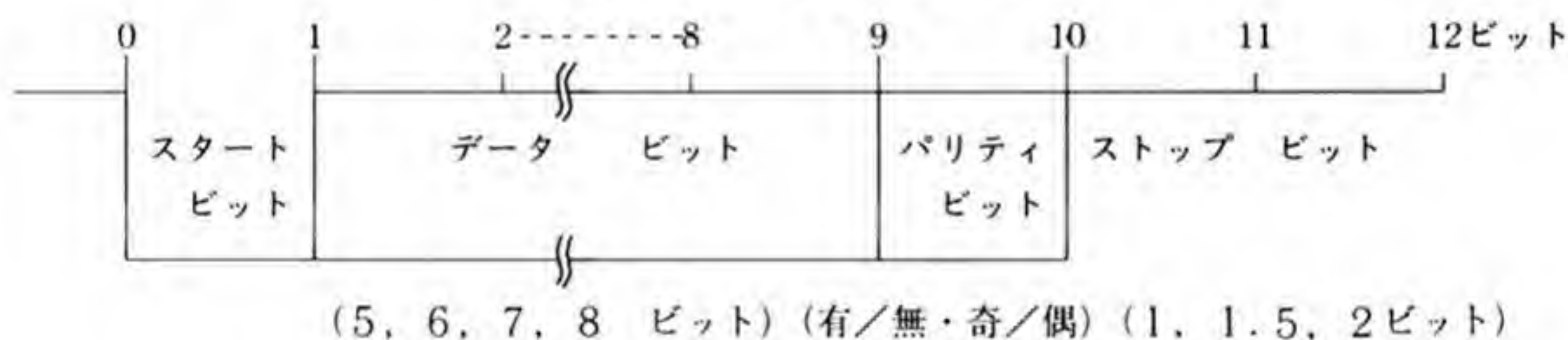



図7　シリアルデータ　フォーマット

3　BASICでRS-232Cインターフェイスを取り扱う方法

RS－232Cインターフェイスで扱うデータはファイルという概念で取り扱い、BASICの入出力命令を使って、一般の入出力装置として使用します。RS－232Cインターフェイスのファイルディスクリプタは次の形式で表わされます。

"COM：通信パラメータ"

通信パラメータは次のとおりです。

〈ボーレート〉〈パリティ〉〈データビット長〉〈ストップビット長〉〈通信制御指定〉〈カナの表現方法指定〉〈CR、LFコードの送信処理〉〈CR、LFコードの受信処理〉〈日本語文字列の表現方法指定〉〈エンドコード指定〉

〈ボーレート〉

0～6の数値によりボーレートを設定します。

数　　値	０	１	２	３	４	５	６
ボーレート(ボー)	１５０	３００	６００	１２００	２４００	４８００	９６００

〈パリティ〉

Ｅ………偶数パリティ・チェックを使用します。

Ｏ………奇数パリティ・チェックを使用します。

Ｎ………パリティ・チェックを使用しません。

〈データビット長〉

５………データビット長を５ビットとします。

６………データビット長を６ビットとします。

７………データビット長を７ビットとします。

８………データビット長を８ビットとします。

〈ストップビット長〉

１………ストップビット長を１ビットとします。

２………ストップビット長を１.５ビットとします。

３………ストップビット長を２ビットとします。

〈通信制御指定〉

通信制御方法を選択します。

Ｘ………ＸＯＮ／ＸＯＦＦコードによる制御を行ないます。

Ｒ………ＲＴＳ制御信号のＯＮ／ＯＦＦによる制御を行ないます。

{Ｎ
省略}…どちらの制御も行ないません。

この通信制御とは、データ送受信時に受信側においてデータの受信処理が間に合わないときに、送信側に対して送信の一時停止を要求し、受信できる状態になると送信再開を要求することです。Ｘを指定するとデータとしてＸＯＦＦコード（＆Ｈ１３）を送信側へ送り返すことにより送信の一時停止を要求し、ＸＯＮコード（＆Ｈ１１）により送信再開を要求します。また、Ｒを指定すると、ＲＴＳ制御信号線をＯＦＦにすることにより送信の一時停止を要求し、ＯＮにすることにより送信再開を要求します。

〈カナの表現方法指定〉

Ｓ………データ７ビットモードでカナの送受信ができます。

{Ｎ
省略}…データ７ビットモードでカナの送受信ができません。

データ７ビットモードでカナの送受信は、シフトアウトコードＳＯ（＆Ｈ０Ｅ）があるとそれ以後のデータはカナとみなし、シフトインコードＳＩ（＆Ｈ０Ｆ）があるとそれ以後のデータを英数字とみなします。

〈ＣＲ，ＬＦコードの送信処理〉

Ｃ………一連の文字列送信後、ＣＲコード（＆Ｈ０Ｄ）を復帰＋改行コードとして処理します。

{Ｌ
省略}…一連の文字列送信後、ＣＲコード（＆Ｈ０Ｄ）＋ＬＦコード（＆Ｈ０Ａ）を復帰＋改行コードとして送信します。

〈CR　LFコードの受信処理〉

C………1つのCRコード（&H0D）の受信で復帰＋改行コードとして処理します。

{L / 省略}…CRコード（&H0D）のみ受信すると、それはデータとして処理され、CRコード（&H0D）とLFコード（&H0A）を連続して受信すると復帰＋改行コードとして処理します。

〈日本語文字列の表現方法指定〉

日本語文字列の表現方法を選択します。

J………漢字インコードKI（&H1B4B）で日本語文字列の始まりを示し、漢字アウトコードKO（&H1B48）で日本語文字列の終わりを示します。

{N / 省略}…シフトJIS漢字コードを使用します。

〈エンドコード指定〉

データ転送の終了を判断するためのエンドコードを指定します。エンドコードを設定するとSAVE命令を実行した場合プログラムデータの送信後指定されたエンドコードを送信してSAVE動作を終了します。LOAD命令を実行した場合、指定されたエンドコードを受信した時点でLOAD動作を終了します。RS—232CファイルをOPEN命令によりアウトプットオープンした場合、CLOSE命令の実行によりエンドコードが送信されます。また、インプットオープンした場合、EOF関数はエンドコードを受信すると真の値となります。

このエンドコードとしてコントロールコード（&H00～&H1F）の中から任意の1つを選択できます。パラメータ値としてはキャラクタコード（&H40～&H5F）に対応する文字を使用します（&H60～&H7Fも可)。たとえば、D（またはd）と指定すると、エンドコードとしてCTRL—D（&H04）が使用されます。またこのパラメータを省略するとエンドコードの送受信処理は行ないません。

(注意) 通信パラメータの〈ボーレート〉から〈ストップビット長〉までは省略できませんが、それ以降のパラメータは省略できます。ただし、途中のパラメータを省略して次のパラメータを指定することはできません。

日本語文字の送・受信はデータ長8ビットのときに可能です。ただし、7ビットモードでもSI／SOコード制御及びKI／KOコード制御を使用することにより可能です。この場合のデータ形式は次のとおりです。

SI	KI	漢字	KO	SO	カナ	SI	KI	漢字	KO	SO	

送／受信方向　→

また、KI／KOコード制御にて日本語文字を送・受信する場合は、漢字表示モードKMODE 1に設定してください。

4　入出力命令と割り込み処理

4.1　RS-232Cインターフェイスに関する入出力命令

SAVE"COM:通信パラメータ"

メインメモリ上のBASICプログラムをアスキー形式にて、RS—232Cインターフェイスを介して相手機器へ送信します。

LOAD"COM:通信パラメータ"

相手機器より送られてきたアスキー形式のBASICプログラムをRS—232インターフェイスを介してメインメモリへ読み込みます。

SAVEM"COM:通信パラメータ",〈開始アドレス〉,〈終了アドレス〉

メインメモリ上のマシン語のプログラムをRS—232Cインターフェイスを介して相手機器へ送信します。

この場合の送信データ形式は次のとおりです。

LOADM"COM:通信パラメータ"

相手機器より送られてきたマシン語プログラムをRS—232Cインターフェイスを介してメインメモリへ読み込みます。

この場合送られてきたデータはSAVEM命令の項に示すデータ形式になっている必要があります。

OPEN" $\left\{\begin{array}{c}I\\O\\C\end{array}\right\}$ ",〔#〕ファイル番号,"COM:通信パラメータ"

I………インプットオープン（受信用）

O………アウトプットオープン（送信用）

C………コミュニケーションオープン（送・受信用）

RS—232Cファイルを開き（通信パラメータの設定を行なうこと）、PRINT#、INPUT#命令等でデータの送・受信ができる状態にします。コミュニケーションオープンはRS—232Cファイルについてのみ有効です。コミュニケーションオープンすると同一ファイル番号でデータの送受信ができます。

PRINT#　ファイル番号,〔X_1, X_2, ……〕

X_1、X_2………出力する式または文字列

RS—232Cファイルがアウトプットオープンされている場合、式または文字列を相手機器へ送信します。文字列送信後、通信パラメータの〈CR、LFコードの送信処理〉でCが設定されている場合にはCRコードがまたLが設定されている場合CR＋LFコードが送信されます。

INPUT#　ファイル番号, X_1〔, X_2, ……〕

X_1、X_2………入力したデータを入れる変数

RS—232Cファイルがインプットオープンされている場合、相手機器より受信したデータを変数に読み込みます。

ＬＩＮＰＵＴ＃　ファイル番号，文字変数

ＬＩＮＥ　ＩＮＰＵＴ＃　ファイル番号，文字変数

文字変数………入力したデータを入れる文字型変数

ＲＳ－２３２Ｃファイルがインプットオープンされている場合、相手機器より受信したデータ（２５５文字まで）を文字変数に読み込みます。通信パラメータの〈ＣＲ、ＬＦコードの受信処理〉でＣが設定されているとＣＲコードが、またＬが設定されているとＣＲ＋ＬＦコードがくるとＬＩＮＰＵＴ動作を完了します。

ＷＲＩＴＥ＃　ファイル番号，〔X_1，X_2，……〕

ＰＲＩＮＴ＃文と同様ですが、データの区切りにはカンマ（,）を送信し、文字列データのときはダブルクォーテーション（"）をつけて、詰めて送信します。

ＩＮＰＵＴ＄（n，〔＃〕ファイル番号）

n………読み込む文字数

ＲＳ－２３２Ｃファイルがインプットオープンされている場合、相手機器より受信したｎ個の文字がこの関数の値になります。

ＬＯＣ（ファイル番号）

受信バッファ（６４バイト）にたまっている文字数がこの関数の値となります。

ＥＯＦ（ファイル番号）

受信動作開始時は偽の値（０）がこの関数の値となり、相手機器よりエンドコードが送られてくると真の値（－１）がこの関数の値となります。

ＣＬＯＳＥ〔〔＃〕ファイル番号１，〔＃〕ファイル番号２，……〕

ＯＰＥＮによって開かれたファイルを閉じます。ＲＳ－２３２Ｃによる送・受信を終了します。アウトプットオープンされていた場合にはエンドコードを送信します。

ＤＥＶＩＣＥ"ＣＯＭ："

デバイス名を省略をしたときに用いられるデバイス名を"ＣＯＭ："に設定します。ＬＯＡＤ、ＳＡＶＥ等のコマンドにおいてデバイス名を省略すると、デバイス名は"ＣＯＭ："と判断します。

（使用例）ＤＥＶＩＣＥ"ＣＯＭ："↵
　　　　　ＳＡＶＥ"６Ｎ８３ＸＮＬＬＪＤ"↵
　　　　　の命令を実行するとＲＳ－２３２Ｃポートへプログラムを送出します。

4.2　割り込み処理

ＩＮＰＵＴ＃、ＩＮＰＵＴ＄命令を実行すると、受信データ待ち状態となり、データを受信するまでプログラムは停止します。　ＯＮ　ＣＯＭ　ＧＯＳＵＢ命令を使用することにより、この受信待ち時間には他の仕事をさせておき、データを受信した時点で指定された行番号からの処理ルーチンの実行を開始することができます。この処理ルーチンからの復帰はＲＥＴＵＲＮ命令によって行なわれます。

ＯＮ　ＣＯＭ　ＧＯＳＵＢ　｛行番号｜"ラベル名"｝

ＲＥＴＵＲＮ　〔｛行番号｜"ラベル名"｝〕

行番号、ラベル名を省略すると、中断した所から処理を再開します。

また、ＣＯＭ　ＯＮ／ＯＦＦ／ＳＴＯＰ命令によりＲＳ－２３２Ｃからの割り込みの許可、禁止、停止を設定できます。

ＣＯＭ　ＯＮ　………割り込み許可
ＣＯＭ　ＯＦＦ　……割り込み禁止
ＣＯＭ　ＳＴＯＰ……割り込み一時保留

サンプルプログラム

ディスクにあるアスキー形式のプログラムを読み込みＲＳ－２３２Ｃインターフェイスより送信します。

```
10 INPUT "File name:",F$
20 F$=LEFT$(F$+SPACE$(13),13)
30 OPEN"I",#1,"1:"+F$
40 OPEN"O",#2,"COM:6N83XNCCND"
50 IF EOF(1)=-1 THEN 100
60 LINPUT #1,D$
70 PRINT D$
80 PRINT#2,D$
90 GOTO 50
100 CLOSE #1,#2
110 END
```

ＲＳ－２３２Ｃインターフェイスより受信したプログラムをグラフィックメモリに書き込みます。

```
10 INPUT "File name:",F$
20 F$=LEFT$(F$+SPACE$(13),13)
30 OPTIONSCREEN 4
40 INIT"MEM:"
50 OPEN"O",#1,"MEM:"+F$
60 OPEN"I",#2,"COM:6N83XNCCND"
70 IF EOF(2)=-1 THEN 120
80 LINPUT #2,D$
90 PRINT D$
100 PRINT#1,D$
110 GOTO 70
120 CLOSE #1,#2
130 END
```

ライン命令実行中にＲＳ－２３２Ｃインターフェイスよりデータを受信すると割り込みがかかり、そのデータを画面に表示して前の処理を再開します。

```
10 INIT:CLS4
20 OPEN"I",#1,"COM:6N83XN"
30 ON COM GOSUB 100
40 COM ON
50 FOR I=0 TO 99
60 LINE(0,100+I)-(319,199),PSET,I,BF
70 NEXT
80 COM OFF:CLOSE #1
90 END
100 PRINT INPUT$(LOC(1),1);
110 RETURN
```

5 ユーザーがマシン語でＲＳ－２３２Ｃ用ソフトを作成する場合の使用法

図８にＲＳ－２３２Ｃ周辺のシステム図を示します。データ伝送におけるボーレート（伝送速度）を決定する基本クロックの作成には、ＣＴＣ（カウンタ／タイマー用ＬＳＩ）を使用し、シリアルデータ通信用ＬＳＩにはＳＩＯを使用しています。ＲＳ－２３２Ｃインターフェイスを使用してデータ通信を行なう場合にはこのＣＴＣとＳＩＯを通信形体に合わせて設定します。以下、ＣＴＣ並びにＳＩＯの設定方法について説明します。

(1) ＣＴＣ（Ｚ－８０Ａ　ＣＴＣ）の設定

ＣＴＣはカウンタ／タイマー機能を持ち、内部に４個の独立したチャンネル（チャンネル０～３）を持っています。このうちチャンネル１と２の出力をＳＩＯに入力し、データ通信におけるボーレートを決定する基本クロックとしています。（ただし、チャンネル２の出力はマウス用に使用しているのでＲＳ－２３２Ｃインターフェイスで使用するのはチャンネル１の出力のみです）ＣＴＣの各チャンネルのＩ／Ｏアドレスは次のとおりです。

チャンネル０………１ＦＡ０（１６進数）
チャンネル１………１ＦＡ１
チャンネル２………１ＦＡ２
チャンネル３………１ＦＡ３

ＣＴＣはカウンタまたはタイマーモードとして使用でき、モードにより分周機能が異なります。ボーレート決定用基本クロックを作るにはカウンタモードで使用し、２ＭＨｚ基本クロックを１／１～１／２５６まで分周できます。

ＣＴＣを動作させるにはまず、チャンネル制御レジスタにそのチャンネルの使用モード等を決定する制御語（ここでは４５（１６進数））を設定し、つづいて時間定数レジスタに分周比として０～ＦＦ（１６進数）を設定します。

```
LD   BC, 1FA1H
LD   A, 45H
OUT  (C), A
LD   A, 0DH
OUT  (C), A
```

これによりＣＴＣのチャンネル１がカウントを開始し、ボーレート決定用基本クロックをＳＩＯへ出力します。ボーレートと時間定数レジスタ値とは一義的には定まらずＳＩＯの設定により変わります。これについて次のＳＩＯの項で説明します。

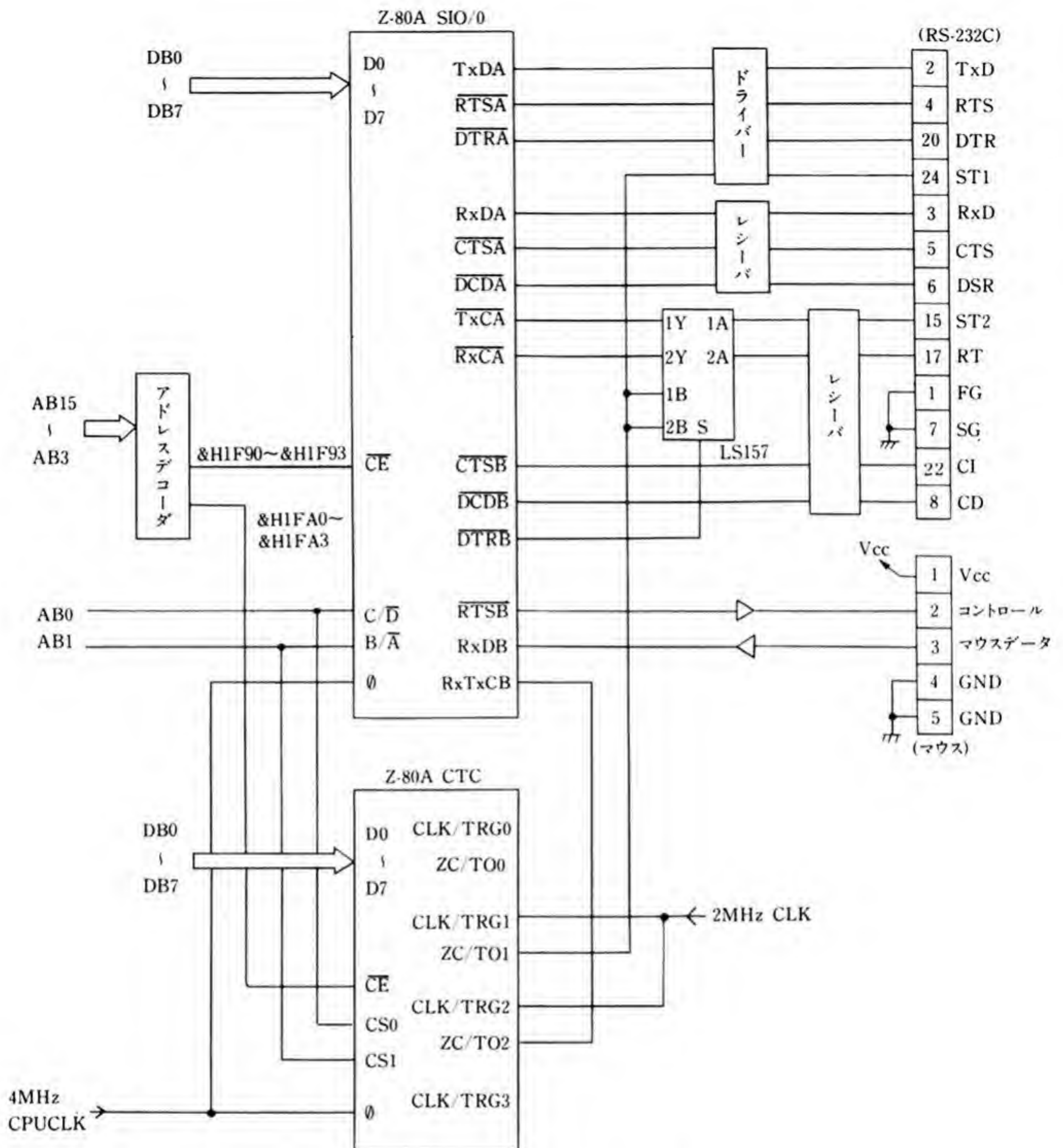



図8．ＲＳ-232Ｃ周辺回路図

(2) ＳＩＯ（Ｚ－８ØＡ　ＳＩＯ／Ø）の設定

ＳＩＯはシリアルデータ通信用チャンネルを２つ（チャンネルＡ、Ｂ）持っており、チャンネルＡをＲＳ－２３２Ｃインターフェイス用に、またチャンネルＢをマウス制御用に使用しています。ＳＩＯを動作させるにはまずコマンド書き込みにより通信モードを設定し、ＳＩＯを初期設定した後データの送・受信を行ないます。ＳＩＯのチャンネル、コマンド／データに対するＩ／Ｏアドレスは次のとおりです。

チャンネルAのデータ………1F90（16進数）
チャンネルAのコマンド……1F91
チャンネルBのデータ………1F92
チャンネルBのコマンド……1F93

SIOにはコマンドの書き込みレジスタを7つ持っています。これらのレジスタにコマンドを書き込む場合には、まず書き込みレジスタ0により実際に書き込みたいレジスタの番号を指定します。これによりそのレジスタへの書き込みが可能となりコマンドを書き込みます。たとえば書き込みレジスタ4にコマンドを書き込む場合は

```
LD  A, 04H
LD  BC, 1F91H
OUT (C), A
LD  A, data……… (コマンドデータ)
OUT (C), A
```

のようにします。この操作を必要とする各レジスタについて行なうことによりSIOは初期化されデータの送・受信が可能となります。データを送・受信する場合には次のようにします。

```
(送信)  LD  A, data……… (送信データ)
        LD  BC, 1F90
        OUT (C), A
(受信)  LD  BC, 1F90
        IN  A, (C) ……… (アキュムレータに受信データが入る)
```

SIOの書き込みレジスタ4の上位2ビット（D_7, D_6）でクロック・レートの設定を行ないます。クロック・レートとは通信ボーレートと送信・受信クロック（$\overline{TxC}$, $\overline{RxC}$）の比率を示し

送信・受信クロック＝ボーレート×クロック・レート
（1, 16, 32, 64）

という関係があります。ただし、クロック・レート1は外部同期にて使用します。

CTC, SIOの設定と通信ボーレートとの関係を表2に示します。

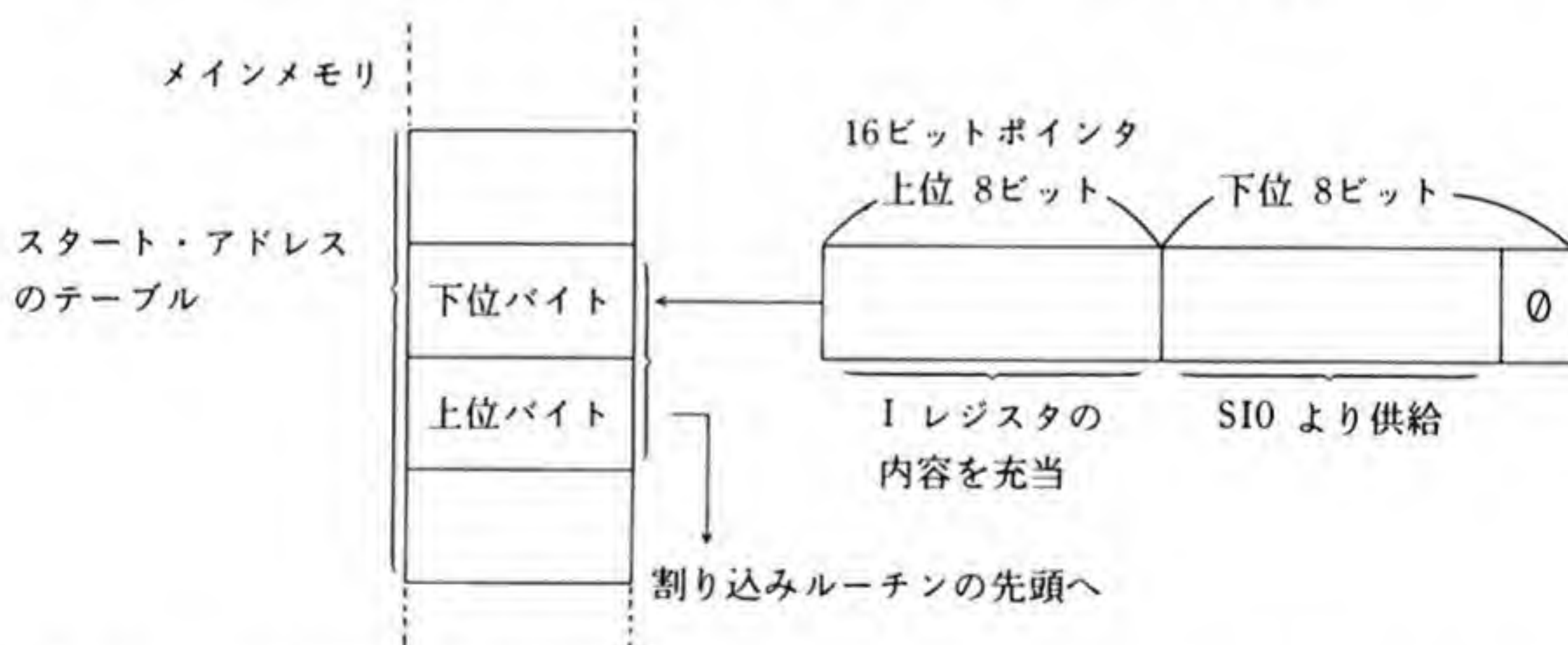
CTCの分周比(時間定数レジスタ値)	SIOのクロック・レートに対するボーレート(ボー)		
	×16	×32	×64
13	9600	4800	2400
26	4800	2400	1200
52	2400	1200	600
104	1200	600	300
208	600	300	150

表2　CTC、SIOの設定とボーレートとの関係

ＲＳ－２３２Ｃインターフェイスを使用してデータの送・受を行なう場合、一般的にはデータ信号線（ＴｘＤ，ＲｘＤ）、コントロール信号線（ＲＴＳ，ＣＴＳ，ＤＴＲ，ＤＳＲ）とＧＮＤ線を使用します。これらの信号線はＳＩＯのチャンネルＡによりすべて制御できます。本機はこれらの信号線の他にＣＩ（被呼表示）、ＣＤ（キャリア検出）を用意しています。この２つの信号線の状態はＳＩＯのチャンネルＢの入力ポートにより読み出せます。また、本機は内部同期だけでなく、外部同期にも対応できるように、ＳＴ１（送信信号エレメントタイミング）、ＳＴ２（送信信号エレメントタイミング）、ＲＴ（受信信号エレメントタイミング）も用意しています。内部／外部同期の切り換えはＳＩＯのチャンネルＢのＤＴＲＢ出力ポートとのＨｉｇｈ／Ｌｏｗにより行ないます。ＳＩＯのチャンネルＢはマウス用に使用しているため、ＤＴＲＢ出力ポートをＨｉｇｈ／Ｌｏｗに切り換える時にＲＴＳＢ出力ポートの状態が変化しないように注意してください。

6　割り込み処理の使用

ＣＰＵとＳＩＯの間で割り込み処理（ベクトル割り込み）を使用する場合について説明します。

この割り込み処理を使用する場合、まえもってプログラムにより割り込みルーチンのスタート・アドレス（２バイト）のテーブルをメモリの適当な位置（偶数アドレスから下位バイト、上位バイトの順）に配置しておきます。ＣＰＵは割り込みを受け付けたとき１６ビットのポインタで、必要な割り込みルーチンのスタート・アドレスをプログラムカウンタに読み込み、そのアドレスへジャンプします。このポインタ（割り込みスタート・アドレスのある位置を示している）として上位８ビットにはＩレジスタの内容を充当し、下位８ビットは割り込みアクノリッジ期間にＳＩＯより割り込みベクトル（プログラムにより設定されていること）が自動的に供給されます。

本機のＢＡＳＩＣを起動すると、Ｉレジスタには＆ＨＦ８が設定されＦ８３０～Ｆ８３Ｂ番地を割り込みスタート・アドレスのテーブルとして使用できます。

14章
マウスインターフェイスの使い方

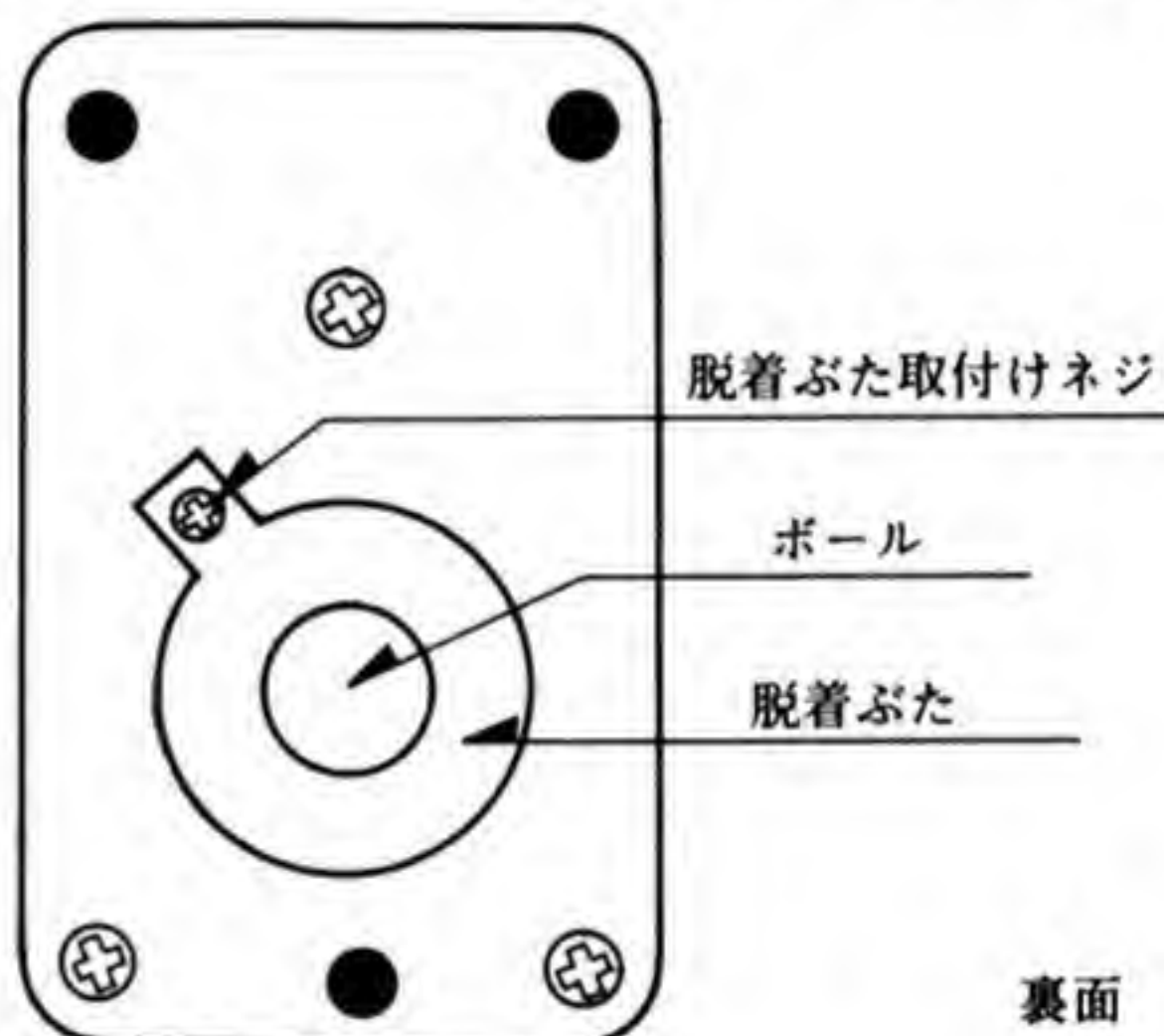
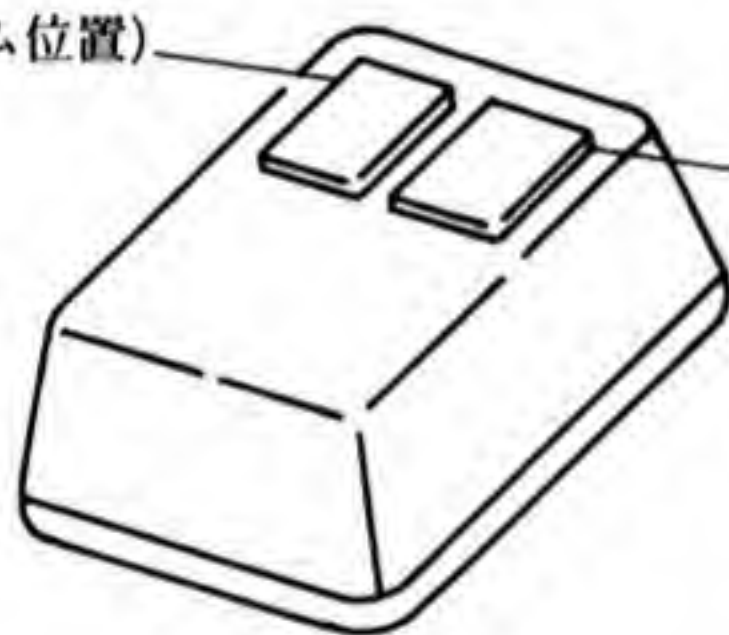
本機ではマウスインターフェイスが内蔵されており、コネクタもコンピュータ本体の前面トビラ内と後面の２ヶ所に用意されています。

1 マウスとは

本機のマウスのハードウェアは、機械式に属します。

マウス裏面（ボタンのついた面の反対面）の中央に、大きな金属のボールがあり、そのボールがマウスの移動により回転し、その回転が本体内のローラに伝わり回転させます。そしてローラは、ロータリーエンコーダに回転を伝えます。これにより、ロータリーエンコーダではマウスの運転量に応じた信号を発生します。

表面

裏面

マウスは、ローラとロータリーエンコーダがＸ軸（横）、Ｙ軸（たて）方向用に１組ずつ用意されています。したがって、マウスの移動量が、Ｘ方向成分、Ｙ方向成分に分けられ、Ｘ、Ｙ座標の相対座標を表わして、そのため座標入力装置〈ポインティング　デバイス〉とも言われます。

また、マウスには２つのボタンがついています。

このボタンは、機能の選択や指示などのセレクト用としてよく使われます。また、ＢＡＳＩＣでもマウス関数でサポートしています。ゲーム用として使う場合にはジョイスティックのトリガーボタンと同じ用途にも使えます。

2 マウスを使う

本機では、ディスク　ＢＡＳＩＣに、マウス関数が用意されています。以後この関数の使い方を中心に説明します。

2.1 マウス関数の書式と機能について

a. 機能の設定ステートメント

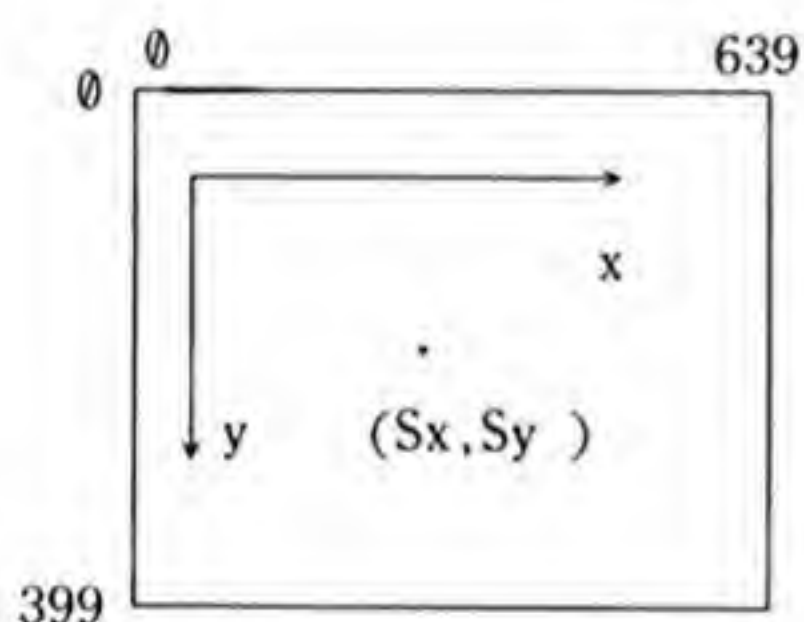
(1)

```
MOUSE 0
```

マウス機能をOFF状態にします。マウスを使用しないのにマウス機能をON状態にしておくと、割り込み注1)の関係で処理スピードが遅くなります。ですからマウスを使わない時は、このステートメントを実行してください。

注1) ここでいう割り込みとは、ある仕事をしていてその途中で別の仕事をし、また元の仕事に戻る、といったことを指します。すなわちマウス機能がON状態だとすると、マウスを使う、使わないに関係なくマウスへの割り込みが発生するということです。別の実行をしているとき途中でマウスの割り込み処理が入るために実行が止まり、その分時間がかかってしまうのです。

(2)

```
MOUSE1, Sx, Sy
```

マウス機能をON状態にし、初期座標として、画面座標(Sx, Sy)を設定します。Sx、Syの値は、−32768〜65535までの値が設定できます。しかし、グラフィック画面上への表示などを考えた場合、WIDTHステートメントによって設定した解像度と一致するように、設定するとよいでしょう。ただし、WIDTHステートメントの設定はOPTION SCREENステートメントによってエラーとなりますので、くわしくは『BASICリファレンスマニュアル』を参照してください。

Sxで設定された値がMOUSE関数のMOUSE(0)に、Syで設定された値がMOUSE(1)の初期値としてセットされます。

Sx、Syを省略するとマウス機能をON状態にするだけとなります。ただし、Sx、Syを省略した場合、BASIC起動時では、MOUSE(0)、MOUSE(1)には、それぞれ0が入ります。

また、Sx、Syを設定した場合、マウスカーソルの移動範囲の設定((4) MOUSE3の項参照)で、その値を超えていた時、移動範囲の最大値に、小さければ最小値に変わります。

(3)

```
MOUSE 2, d, r
```

X(横)方向、Y(たて)方向のマウスカーソルの移動比率を設定します。dは、マウスカーソルの移動方向を指し、d=0では、X(横)方向、d=1では、Y(たて)方向の移動比率の設定となります。移動比率rは、1〜32の値を設定できます。rの値が大きくなるほど、マウスの移動に対する座標の変化は小さくなります。それは、マウスの移動量(マウスから送られてくる値)を移動比率で割っているためです。つまりrの値が大きくなるほどMOUSE(0)、MOUSE(1)、に加えられる値が小さくなり、座標の変化が小さくなります。

BASIC起動時では、X、Y方向の移動比率rはそれぞれ10の値です。

(4)

```
MOUSE  3, Sx1, Sy1, Sx2, Sy2
```

マウスカーソルの移動範囲を設定します。

座標（Sx_1，Sy_1）と（Sx_2，Sy_2）を結ぶ線を対角線とする四角形で囲まれた領域をカーソル移動範囲とします。グラフィック画面との関係からＷＩＤＴＨステートメントによって設定した解像度と一致させるとよいでしょう。

『ＢＡＳＩＣリファレンスマニュアル』のＷＩＤＴＨ、ＯＰＴＩＯＮ　ＳＣＲＥＥＮを参照してください。

また、(2)の項のＭＯＵＳＥ１，Ｓｘ，Ｓｙで設定された座標（Ｓｘ，Ｓｙ）が、この移動範囲からずれていた場合、移動範囲にいちばん近い座標となります。実際は、移動範囲を出ていた方の値が移動範囲の最大値または最小値となります。

座標（Sx_1，Sy_1）と（Sx_2，Sy_2）を設定するときは、$Sx_1 < Sx_2$，$Sy_1 < Sy_2$でなくてはなりません。

b. 状態の読み出し関数

(1)

```
MOUSE (0)
```

マウスカーソルの現在のＸ（横）座標をこの値として返します。

(2)

```
MOUSE (1)
```

マウスカーソルの現在のＹ（たて）座標をこの関数の値として返します。

(3)

```
MOUSE (2, b)
```

指定されたボタン番号ｂのボタンが押されているときに、－１の値をこの関数の値として返し、押されていないときは、０の値を返します。

ボタン番号ｂとは、マウスのケーブルの出ている方向を上にして、

　　ｂ＝１で左側のボタンが押されている時のみ関数は－１

　　ｂ＝２で右側のボタンが押されている時のみ関数は－１

となり条件を満たさない場合この関数は０となります。

(4)

```
MOUSE (3, b)
```

ボタン番号ｂのボタンが押された時のマウスカーソルのＸ（横）座標をこの関数の値として返します。

(5)

```
MOUSE (4, b)
```

ボタン番号bのボタンが押された時のマウスカーソルのY（たて）座標をこの関数の値として返します。

(6)

```
MOUSE (5, b)
```

ボタン番号bのボタンが離された時のマウスカーソルのX（横）座標を、この関数の値として返します。もちろんボタンは、押されていた必要があります。

(7)

```
MOUSE (6, b)
```

ボタン番号bのボタンが離された時のマウスカーソルのY（たて）座標をこの関数の値として返します。

(8)

```
MOUSE (7)
```

最後にこの関数が呼び出されてから、次にこの関数が呼び出されるまでにマウスカーソルが動いたX（横）方向の移動距離をこの関数の値として返します。つまり、最後にこの関数が呼び出された時の座標と、次のこの関数が呼び出された時の座標のX座標の差がこの関数の値となります。

(9)

```
MOUSE (8)
```

最後にこの関数が呼び出されてから、次にこの関数が呼び出されるまでにマウスカーソルが動いたY（たて）方向の移動距離をこの関数の値として返します。つまり最後にこの関数が呼び出された時のマウスカーソルの座標と次にこの関数が呼び出された時のマウスカーソルの座標のY座標の差がこの関数の値となります。

2.2 マウス関数を動かす

次のプログラムを入力してみてください。

```
10 INIT:WIDTH 80,25,0,0
20 MOUSE1,0,0
30 MOUSE2,0,5
40 MOUSE2,1,12
50 MOUSE3,0,0,639,199
60 X=MOUSE(0):Y=MOUSE(1)
70 S1=MOUSE(2,1):S2=MOUSE(2,2)
80 PRINT "X= ";X,"Y= ";Y,"SW1= ";S1,"SW2= ";S2
90 GOTO 60
```

ＲＵＮ⏎してみますと、下の様に表示されます。

Ｘ＝　　0　　Ｙ＝　　0　　ＳＷ２＝　　0

ここで、マウスを動かしてみてください。

Ｘ＝　とＹ＝　の後の数値が変わります。

ではボタンを押したり離したりして見てください。

左側のボタンを押すと、「ＳＷ１＝　－１」の表示が、右側のボタンを押すと、「ＳＷ２＝　－１」の表示が現われ、離すとそれぞれ－１が0になることがわかります。

では、プログラムを止めてＬＩＳＴ⏎を入力してください。

プログラムの止め方は、SHIFT＋BREAKキーとしてください。

プログラムの説明をします。

行番号１0は、画面を初期化し、画面座標系を６４0×２00ドットの画面に設定します。専用ディスプレイテレビ以外のモニターをお使いの方は本機の標準／高解像動切換スイッチで合わせてください。プログラムを直す必要はありません。

行番号２0は、マウス機能をＯＮ状態にし、初期座標を（0，0）に設定しますので最初は、Ｘ＝0、Ｙ＝0を表示します。

行番号３0は、Ｘ（横）方向の移動比率を５と設定します。

行番号４0は、Ｙ（たて）方向の移動比率を１２と設定します。

行番号３0と４0の移動比率（１～３２の値）をいろいろ変えてみてください。違いがわかります。

行番号５0は、マウスカーソルの移動範囲を（0，0）－（６３９，１９９）に設定します。プログラムを実行させた時、マウスを動かしてみてください。移動の範囲の値しかとれません。大きくなったり、小さくなったりはしません。

行番号６0は、変数Ｘにマウスカーソルのｘ(横)座標の値を、変数Ｙにマウスカーソルのｙ(たて)座標の値を代入します。

行番号７0は、変数Ｓ１に左側のボタンの状態を変数Ｓ２に右側のボタンの状態を代入します。どちらもボタンが押されていると－１の値が、離されているとき0の値が入ります。

行番号８0は、それぞれの変数と変数の値の表示をします。

行番号９0は、６0にジャンプします。

2.3　マウスカーソルを表示させる

本機ではマウスカーソルを表示するには、そのためのプログラムの入力が必要です。次に示すのがそのサンプルです。図１の様なエンピツ状のマウスカーソルを表示し、マウスの左のボタンを押しながらマウスを動かすと線を引き右のボタンで画面をクリアするというものです。

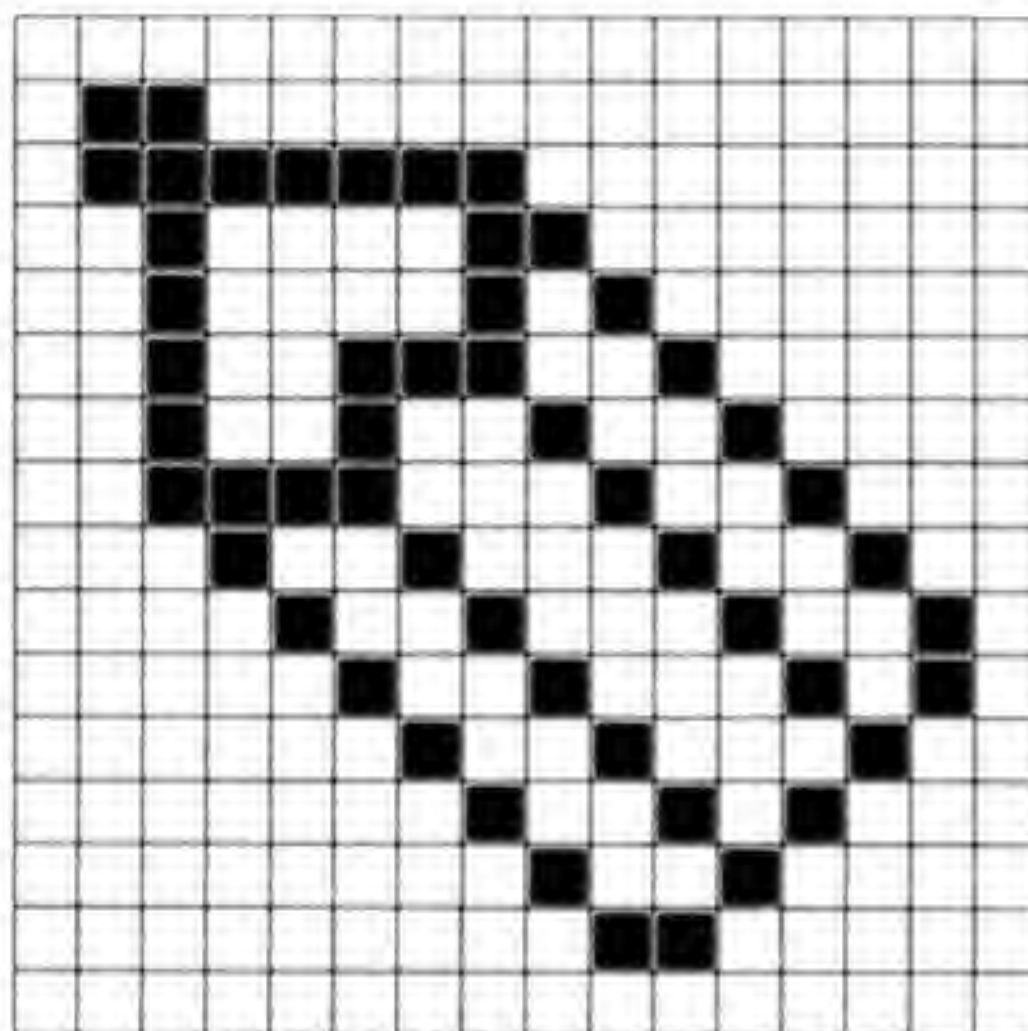
```
10 INIT:PALET:OPTIONSCREEN0
20 WIDTH 80,25,1,2
30 LINE(0,0)-(7,7),XOR,BF,HEXCHR$("000000606060 7F7F7F 212121 212121 272727 242424 3C3C3C")
40 LINE(8,0)-(15,7),XOR,BF,HEXCHR$("000000 000000 000000 808080 404040 202020 909090 484848")
50 LINE(0,8)-(7,15),XOR,BF,HEXCHR$("121212 090909 040404 020202 010101 000000 000000 000000")
60 LINE(8,8)-(15,15),XOR,BF,HEXCHR$("242424 121212 8A8A8A 444444 282828 909090 606060 000000")
70 DIM A(50)
80 GET@(0,0)-(15,15),A,7
90 X=0:Y=0:XX=0:YY=0
100 MOUSE1,0,0
110 MOUSE2,0,5
120 MOUSE2,1,4
130 MOUSE3,0,0,624,384
140 PUT@(X,Y)-(X+15,Y+15),A,XOR,7
150 X=MOUSE(0):Y=MOUSE(1)
160 IF MOUSE(2,1)=-1 THEN LINE(XX,YY)-(X,Y),PSET
170 PUT@(X,Y)-(X+15,Y+15),A,XOR,7
180 XX=X:YY=Y
190 IF MOUSE(2,2)=-1 THEN CLS0:GOTO 150
200 GOTO 140
```

データ	
００	００
６０	００
７F	００
２１	８０
２１	４０
２７	２０
２４	９０
３C	４８
１２	２４
０９	１２
０４	８A
０２	４４
０１	２８
００	９０
００	６０
００	００

図１　マウスカーソルの形状例とデータ（16進数）

プログラムの説明をします。

行番号１０は、画面の初期化です。

行番号２０は、グラフィック画面の解像度を６４０×４００ドットにします。ですから専用ディスプレイテレビまたは高解像度ディスプレイをお使いでない方はＷＩＤＴＨ　８０，２５，０，０に直してください。ただし、マウスカーソルの形状は画面のたて、横比の関係で少しゆがみます。３０～６０は、マウスカーソルの形状をグラフィック画面の座標（０，０）と（１５，１５）を対角点とする四角形の中に表示します。実際は見えません。

行番号７０は、配列Ａを５１個用意します。

行番号８０は、先ほどグラフィック画面に表示したマウスカーソルの形状を配列Ａに読み込みます。

行番号９０は、変数の初期値を０とします。

行番号１００は、マウス機能をＯＮ状態にし、初期座標を（０，０）に設定します。

行番号１１０は、Ｘ（横）方向の移動比率を５と設定します。

行番号１２０は、Ｙ（たて）方向の移動比率を４と設定します。

行番号１３０は、マウスカーソルの移動範囲（０，０）－（６２４，３８４）に設定します。ここで、注意して頂きたいのは、グラフィック画面の座標（０，０）－（６３９，３９９）と一致していない点です。これは、ＰＵＴ＠ステートメントの有効範囲がこの場合、Ｘ方向６３９、Ｙ方向３９９となっており、行番号１４０のステートメントがＸ＋１５，Ｙ＋１５になっています。マウスの座標Ｘ、Ｙが（６２５，３８５）を超えると、ＰＵＴ＠の有効範囲を超えてエラーとなります。行番号２０でＷＩＤＴＨ　８０，２５，０，０ とした方は、ＭＯＵＳＥ　３，０，０，６２４，１８４としてください。

行番号１４０、１７０は、８０で読み込んだ図形を表示します。

行番号１５０は、ＸにマウスのＸ座標をＹにマウスのＹ座標を代入します。

行番号１６０は、マウスの左のボタンが押されていると線をひきます。

行番号１８０は、ＸＸにＸの値をＹＹにＹの値を代入します。

行番号１９０は、マウスの右のボタンが押されると画面をクリアします。

行番号２００は、１４０にジャンプします。

3　使用上の注意事項

本機にマウスを接続する場合、コンピュータ本体の前面トビラ内と後面の２箇所のコネクタに同時に取り付けて実行することはおやめください。正常な動作をしません。

参　考　グラフィックツール・嬉楽画ターボ・ＣＺ－１１４ＳＦにはマウスが付属されています。内容については「１５章デジタルテロッパーの使い方」の **参考** をお読みください。

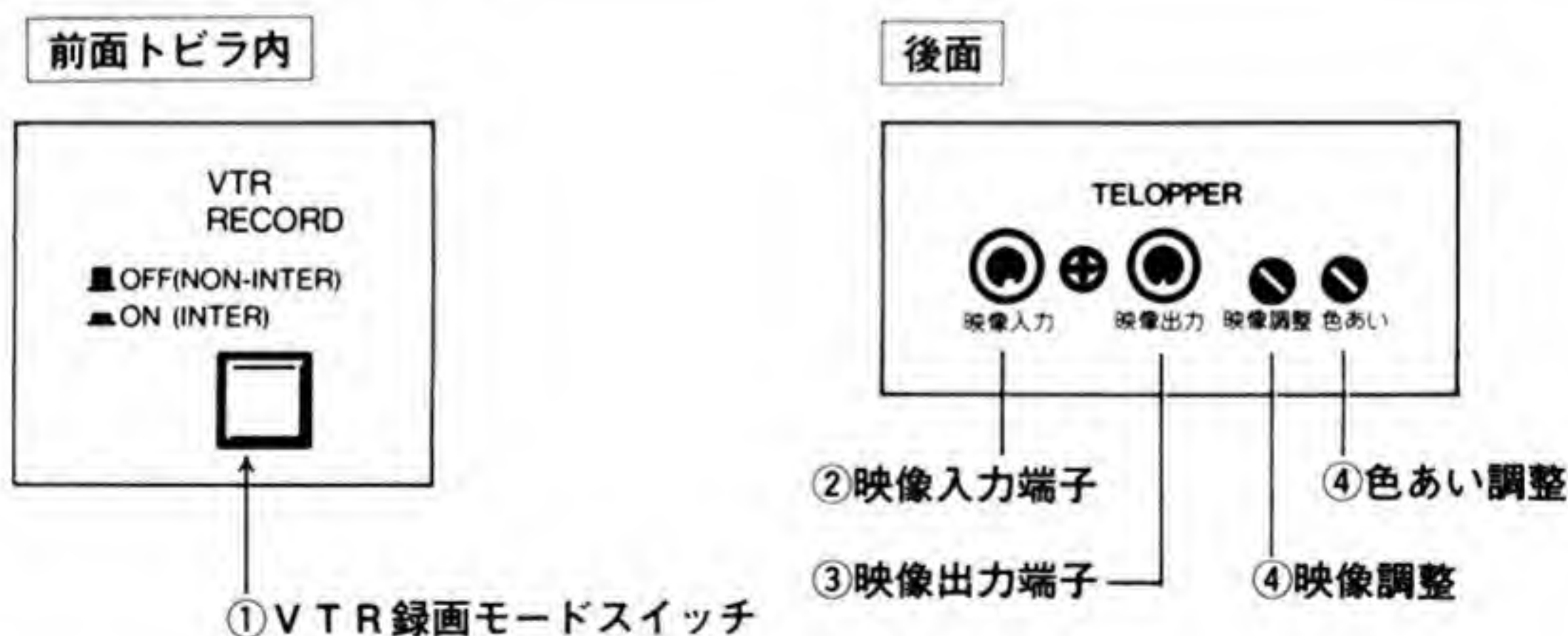
15章
デジタルテロッパーの使い方

本機では、専用ディスプレイテレビとの組み合わせ、あるいは家庭用テレビとの組み合わせにより、コンピュータ画像とテレビやビデオ画像との重ね合わせ〈**スーパーインポーズ**〉**画像**が表示できます。さらに、内蔵の**デジタルテロッパー機能**を利用すれば、コンピュータ画像やスーパーインポーズ画像のビデオ画像の録画が可能になります。

1 特長

スーパーインポーズ処理部にＳＳＳ（セパレート・サブキャリア・スーパーインポーズ）方式を採用しました。

- コンピュータ映像部のエッヂノイズ（輝度、色、フリッカー）が極めて少なく安定高画質なスーパーインポーズを実現しました。
- 入力映像ソースの画像劣化がほとんどありません。
- ＶＴＲ等不安定な映像ソースにも、安定した同期で、かつ、コンピュータ映像レベル、色あいを保持するダブルクランプ方式（ピーク＆ペデスタルクランプ）を採用しました。

2 各部名称

①ＶＴＲ録画モードスイッチ

コンピュータ画像をＶＴＲに録画するときはスイッチをＯＮ（インターレースモード）にします。詳しくは「3 ＶＴＲ録画モードスイッチについて」を参照してください。

②映像入力端子

接続機器（テレビ、ＶＴＲ、ビデオディスク、ビデオカメラなど）からの映像信号を入力する端子です。

③映像出力端子

コンピュータ画像（ＮＴＳＣ信号に変換されたもの)、スーパーインポーズ画像、ビデオ画像の出力を行なう端子です。

④映像調整、色あい調整

映像出力端子より出力されるコンピュータ画像の調整ボリュームですが、あらかじめ、標準状態に設定してあります。調整したい場合は販売店にご相談ください。

3 ＶＴＲ録画モードスイッチ

本機前面のトビラ内にはＶＴＲ録画モードスイッチがありますが、これには２つのモードがあります。以下このモードについて説明します。

このスイッチは、コンピュータ画面にしたとき使用し、その画像を録画する場合にはＯＮにし、録画しない場合はＯＦＦとします。

なお、テレビ（またはビデオ）画面、スーパーインポーズ画面にした場合は、自動的にインターレースモードになります。

	ＶＴＲ録画モードスイッチＯＦＦ(■) (ノンインターレースモード)	ＶＴＲ録画モードスイッチＯＮ(▬) (インターレースモード)
使用目的	•コンピュータ画像をＶＴＲに録画しないとき（プログラム作成時など）はＯＦＦにしておきます。	•コンピュータ画像をＶＴＲに録画するときはＯＮにします。
出力信号形態	•コンピュータ本体内部で発生する同期信号を使って出力するので、ＮＴＳＣ標準複合信号から少しはずれた信号です。(内部同期) •ノンインターレース走査	•コンピュータ本体へ入力する映像信号の同期信号を使って出力するので正規のＮＴＳＣ標準複合信号です（外部同期) •インターレース走査
映像入力	不　要	必　要
特　長	•ノンインターレース走査のため映像の垂直ゆれ（フリッカー）がありません。	•正規のＮＴＳＣ標準複合信号ですからＶＴＲに録画が可能です。 •白文字に着色現象がありません。 •モード切り換え（スーパーインポーズ／コンピュータ）を行なっても同期の乱れがありません。
注意事項	•白文字に着色現象があります。 •くし形フィルター方式のテレビと本機とを接続した場合には、一般のテレビよりもコンピュータ画面に、にじみがでます。	•ＶＴＲを映像入力として使用した場合、ＶＴＲ再生時以外（停止、早送り、巻戻しなど）で画像に乱れが生じることがあります。

4 使用方法

本機内蔵のデジタルテロッパーにより映像出力端子から次の３種類の信号が出力されます。

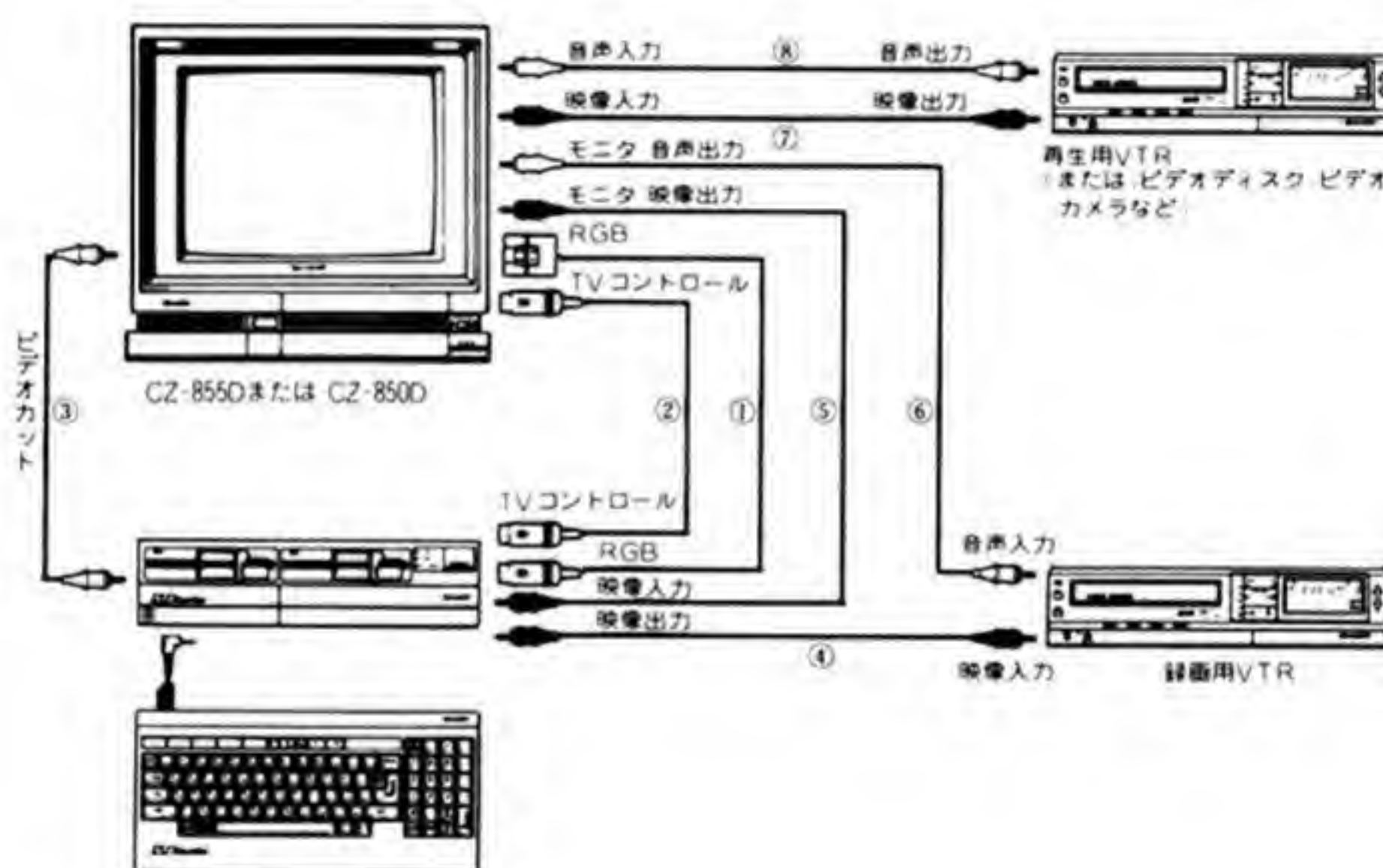
①ＮＴＳＣ信号に変換されたコンピュータ画像の出力。(ノンインターレースモードとインターレースモードの２種類あります。)

②映像入力端子に入力されたＮＴＳＣ信号とコンピュータのＲＧＢ信号を一つのＮＴＳＣ信号に合成・変換したスーパーインポーズ画像の出力。

③映像入力端子に入力されたＮＴＳＣ信号をそのまま出力。

接続については、ご使用になる接続機器（ＶＴＲ、ビデオディスク、ビデオカメラ、ビデオ入出力端子付テレビなど）の組み合わせによってさまざまな接続方法がありますが、ここでは標準的な接続方法の２通りを説明をします。

接続する時には各接続機器の電源をかならずＯＦＦ（切）にしてください。なお接続用ケーブルは最寄りの販売店でお買い求めください。

(1) 専用ディスプレイテレビとの組み合わせ例

接続方法

①本機とディスプレイテレビとをＲＧＢ信号用ケーブル、テレビコントロールケーブル、ビデオカッ
｜
③ト用ケーブルで接続します。

④本機の映像出力端子と録画用ＶＴＲの映像入力端子とを接続します。

⑤本機の映像入力端子とディスプレイテレビのモニター出力端子（映像）とを接続します。

⑥ディスプレイテレビのモニター出力端子（音声）と録画用ＶＴＲの音声入力端子とを接続します。

⑦ディスプレイテレビの映像入力端子と再生用ＶＴＲの映像出力端子とを接続します。

⑧ディスプレイテレビの音声入力端子と再生用ＶＴＲの音声出力端子とを接続します。

操作方法

接続ができましたら、各接続機器の電源をＯＮ（入）にしてください。なお、各接続機器の取扱いや操作については、それぞれの取扱説明書にしたがってください。

①コンピュータ本体前面トビラ内のＶＴＲ録画モードスイッチをＯＮ（入）にしてください。

②テレビ放送を録画したいときは、ディスプレイテレビＣＺ－８５５ＤまたはＣＺ－８５０Ｄをテレビ画面にし、再生用ＶＴＲの映像を録画したいときは、ビデオ画面（画面表示はＰ.）にします。
（コンピュータ画像　のみを録画したい場合、テレビ画像またはビデオ画像などのＮＴＳＣ信号をコンピュータの映像入力端子に入力する必要があります。）

●コンピュータ画像のモニター録画する場合

本機のキーボードで[SHIFT]キーを押しながらテンキーの[・]キーを押すとディスプレイテレビにはコンピュータ画像が映し出され、録画できる状態になります。

●スーパーインポーズ画像のモニター・録画をする場合

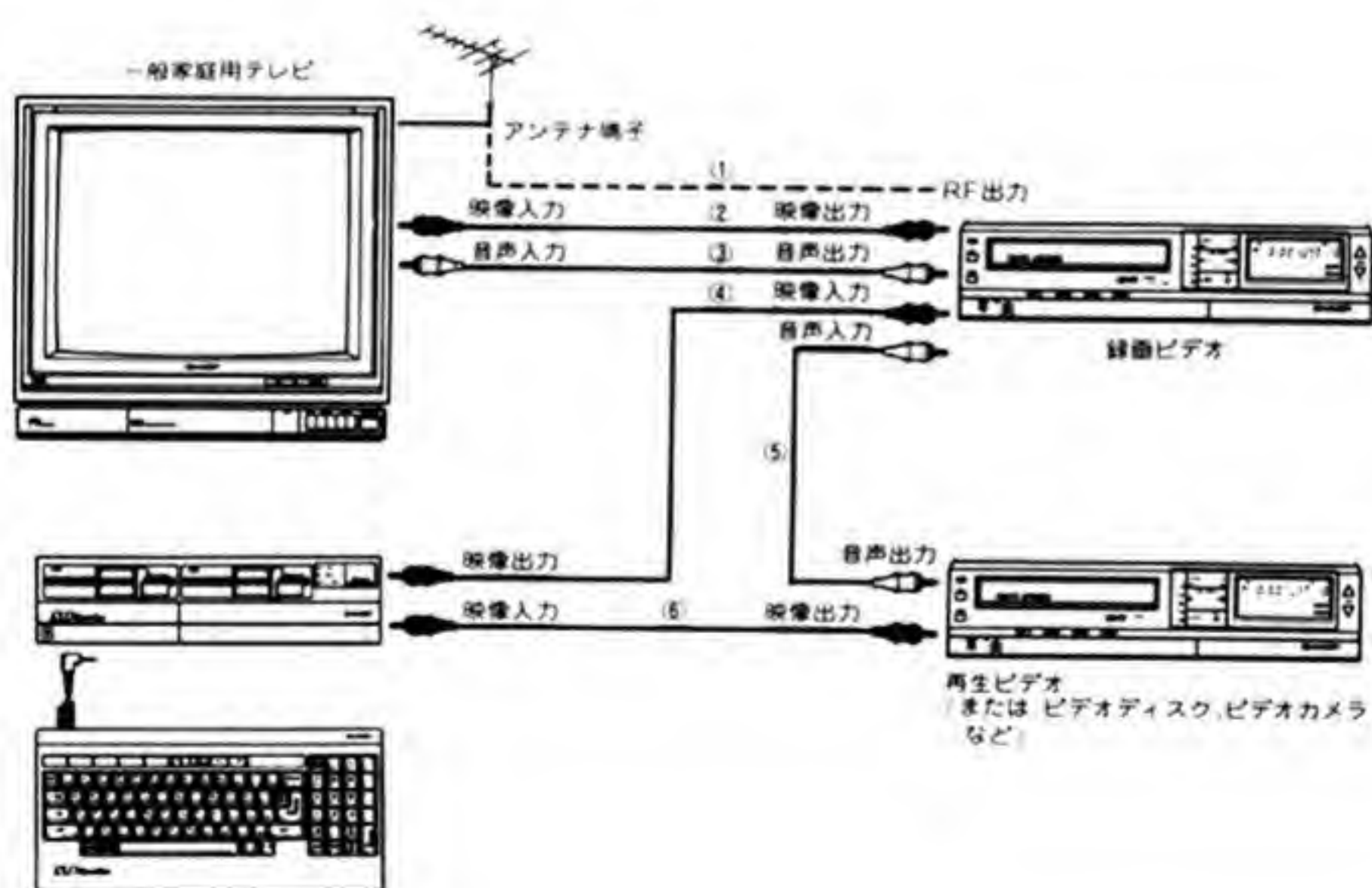
本機のキーボードで[SHIFT]キーを押しながらテンキーの[+]キーを押すとディスプレイテレビにはスーパーインポーズ画像が映し出され、録画できる状態になります。

●ビデオ画像のモニター・録画をする場合

本機のキーボードで[SHIFT]キーを押しながらテンキーの[=]キーを押すとディスプレイテレビにはテレビまたはビデオ画像が映し出され、録画できる状態になります。

●プログラムを組む場合

前面トビラ内のＶＴＲ録画モードスイッチをＯＦＦ（切）にし、本機のキーボードで[SHIFT]キーを押しながらテンキーの[・]を押します。

(2)一般家庭用テレビとの組み合わせ例

＊映像入力端子付テレビの場合は②，③の接続を行ない、端子がない場合には①の接続を行ないます。

接続方法

①テレビに映像入力端子がない場合、この接続を行ないます。録画用ＶＴＲのＲＦ出力とテレビのアンテナ端子を接続します。

②テレビに映像入力端子がある場合、この接続を行ないます。録画用ＶＴＲの映像出力端子とテレビの映像入力端子を接続します。

③テレビに音声入力端子がある場合、この接続を行ないます。録画用ＶＴＲの音声出力端子とテレビの音声入力端子を接続します。

④本機の映像出力端子と録画用ＶＴＲの映像入力端子を接続します。

⑤再生用ＶＴＲの音声出力端子と録画用ＶＴＲの音声入力端子を接続します。

⑥再生用ＶＴＲの映像出力端子と本機の映像入力端子を接続します。

以上の接続ができましたら、各接続機器の電源をＯＮ（入）にして次のように操作してください。
なお、各接続機器の取り扱いについては、各々の取扱説明書にしたがってください。

操作方法

専用ディスプレイテレビとの組み合わせ例の操作方法を参照してください。
ただし、②項については、次に従ってください。

② 接続の手順で、①を行なった場合は、チャンネルを１ｃｈまたは２ｃｈに合わせます。
②③を行なった場合は、ビデオ画面にします。

5 黒抜き表示

本機では、テキスト画面、グラフィック画面ともに黒抜き表示が可能です。つまり、テキスト画面ではグラフィック画面、テレビ画面に対する黒抜き表示が行なえ、グラフィック画面ではテキスト画面、テレビ画面に対する黒抜き表示が行なえます。

黒抜き表示は、専用ディスプレイテレビ（ＣＺ－８５５ＤまたはＣＺ－８５０Ｄ）との組み合わせでは双方のビデオカット端子を接続することによって　ＲＧＢ出力で直接行なうことができます。それ以外のモニターでは本機の映像出力端子から出力されるＮＴＳＣ信号によって、黒抜き表示を行うことができます。

6 注意事項

①このテロッパー機能は標準ディスプレイモードのときのみ使用可能で高解像度ディスプレイモードでは使用できません。

②コンピュータ画像またはスーパーインポーズ画像をＶＴＲに録画する際はＮＴＳＣ信号という帯域制限を受けるため、コンピュータ画面を下のいずれかに設定してください。

４０字×１０行、４０字×１２行、４０字×２０行、４０字×２５行

また、タイリングペイントの様な細いドット構成の画面はちらつき、色変化を生ずることがあります。さらにＶＴＲの性能により録画画像に色ずれ、画質の劣化を生ずることがあります。

（理由）

ＲＧＢ信号の周波数特性は　１０－１４ＭＨｚであるのに対し、ＮＴＳＣ信号は下図に示すように輝度信号の後ろにある色信号と交錯しないようにするため、３～３.２ＭＨｚ程度の帯域巾となっています。このため　横８０字（水平６４０ドット）のように細い指定をしても　見にくかったり、色つきが悪かったりします。

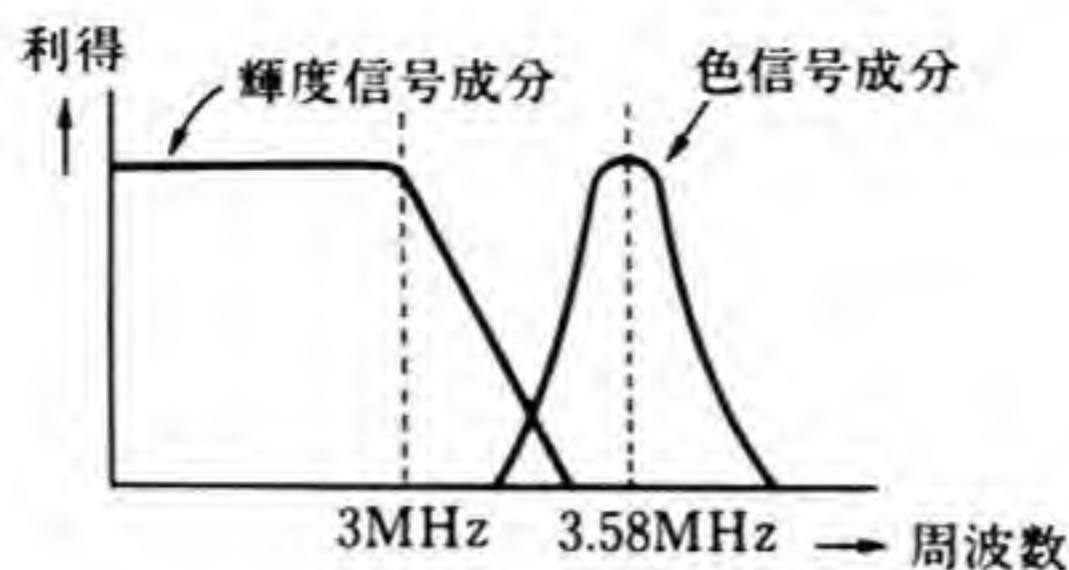
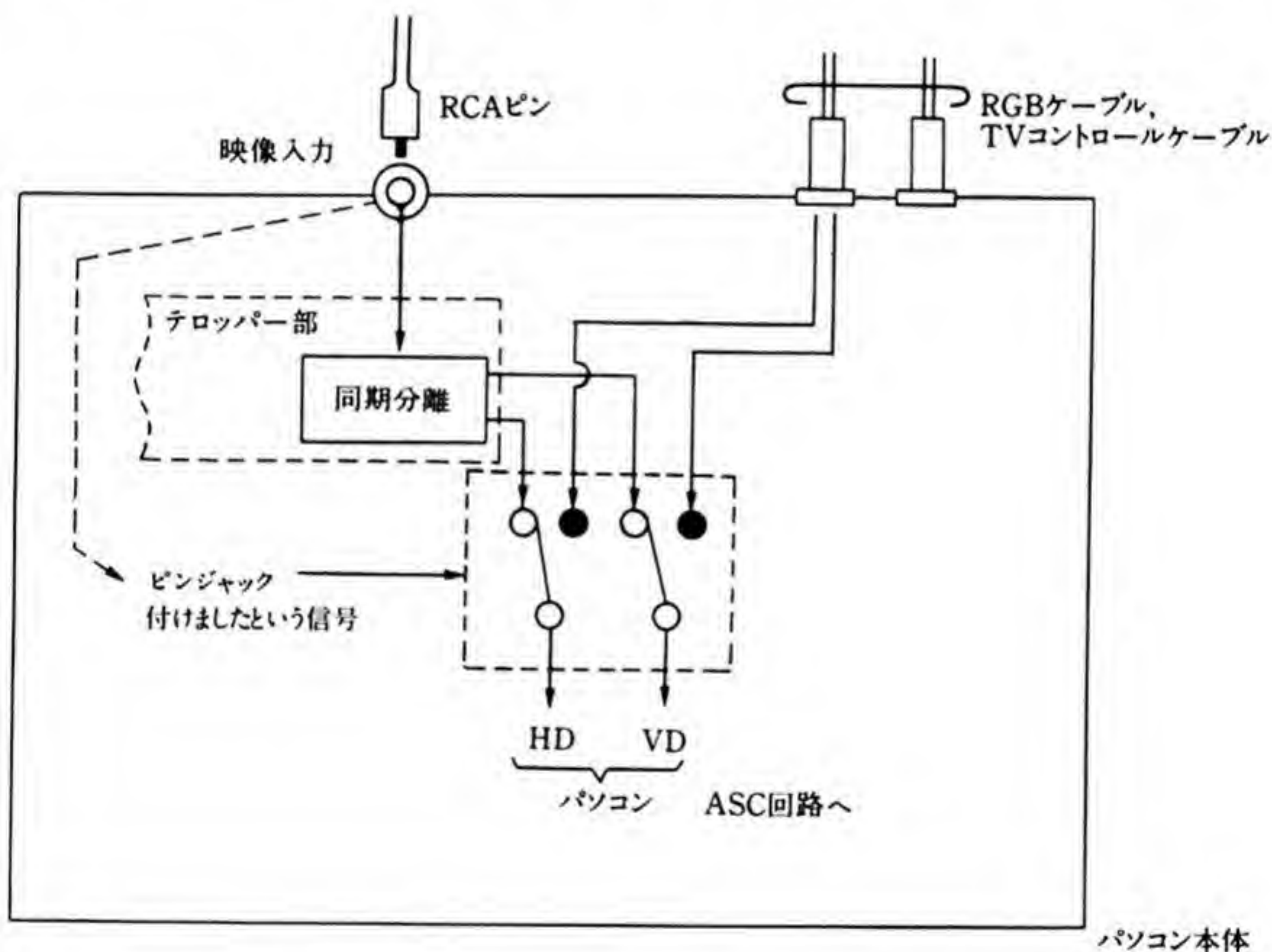



図. NTSC信号の周波数スペクトラム

③本機の映像入力端子にＲＣＡピンケーブルを接続し、そのケーブルが他の映像機器（ＶＴＲ、ビデオカメラ、ビデオディスク等）の映像出力端子に接続されていない場合や、接続されていても、映像信号が本機に入力されている場合、本機の映像出力信号はもとより、モニターの画像も乱れます。従ってデジタルテロッパー使用の際は、必ず映像入力端子に他の機器より映像信号を入力してください。もし、デジタルテロッパーを用いない場合にはＶＴＲ録画モードスイッチをＯＦＦ（切）にするか、もしくは、本機の映像入力端子からＲＣＡピンケーブルをはずしてください。

（理由）
スーパーインポーズは映像信号にＲＧＢ信号を同期させて行なっています。専用ディスプレイテレビと本機との組み合わせでは、ディスプレイテレビの水平同期（ＨＤ）信号、垂直同期（ＶＤ）信号にＲＧＢ信号を同期させてスーパーインポーズを行ないます。一方、外部より映像機器（ＶＴＲ、ビデオディスク、ビデオカメラなど）を接続し、ビデオ画面とのスーパーインポーズを行なう場合、本機の映像入力端子と映像機器の映像出力端子とをＲＣＡピンケーブルで接続しますが、このとき本機内部では、ＲＣＡピンジャックを差し込んだ時から、映像機器からＨＤ、ＶＤ

信号を供給することになります。したがって、本機の映像入力端子にＲＣＡピンジャックを差し込んで、映像機器に接続されないままで放置されると、ＨＤ、ＶＤ信号が本機に入らなくなり、画面がみだれて何も見えなくなります。

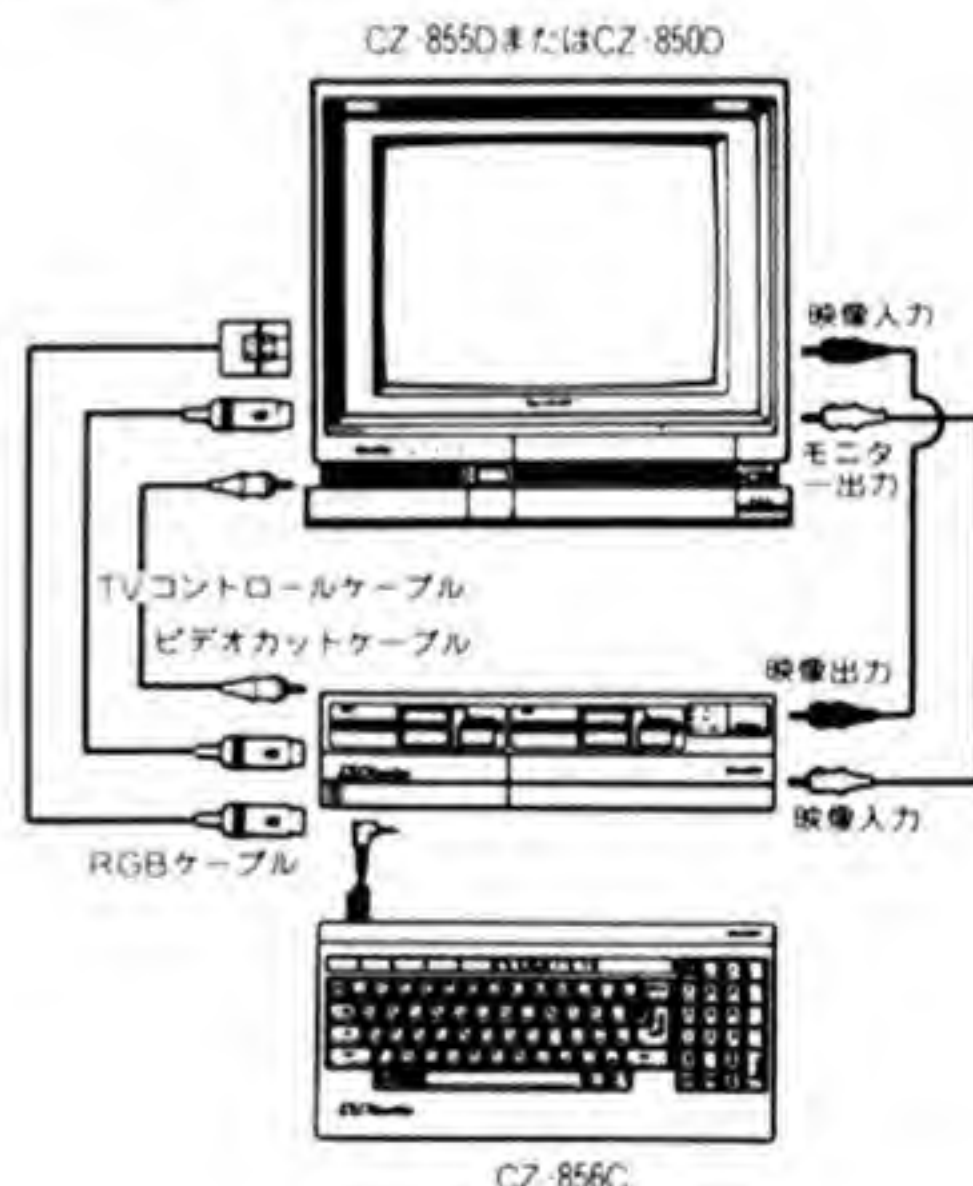
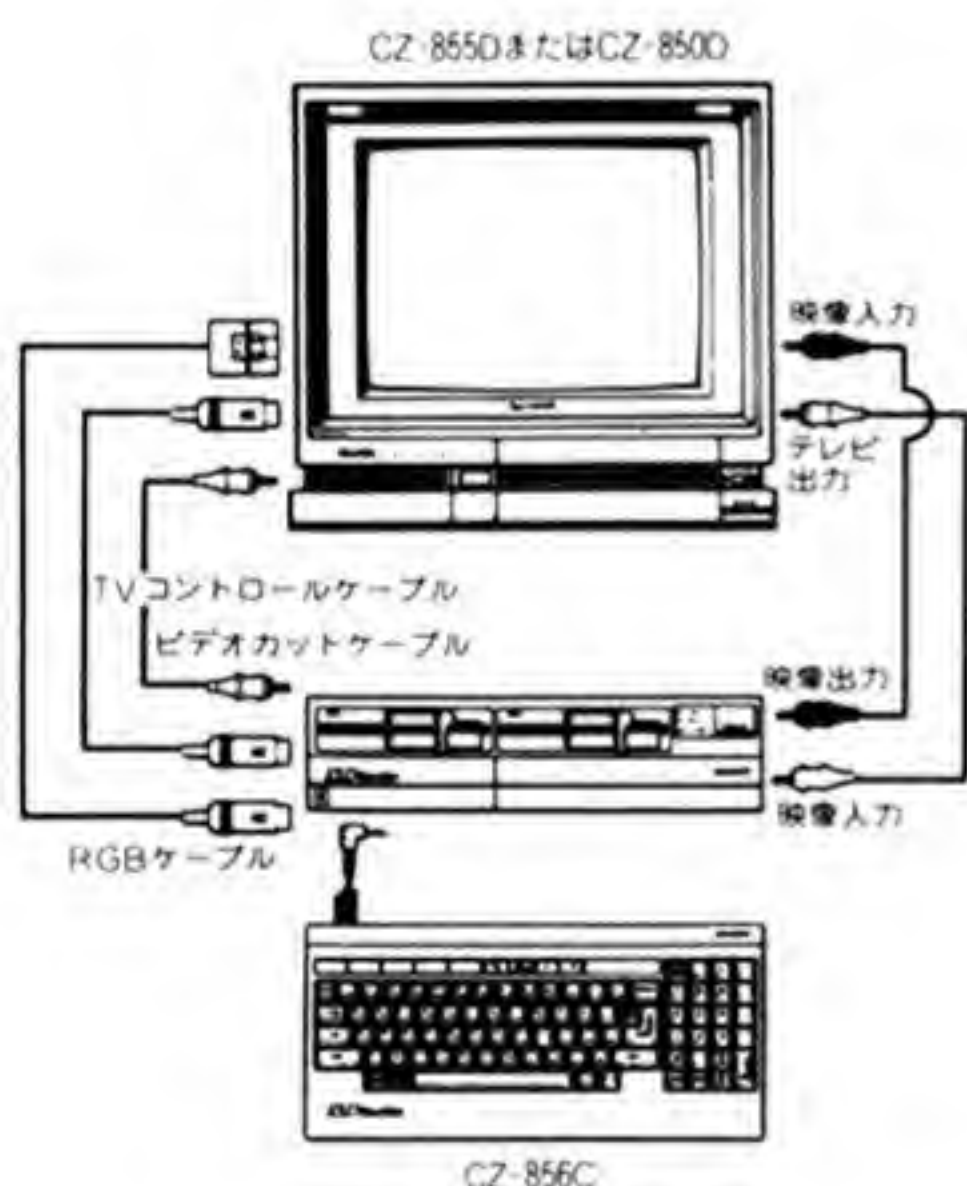
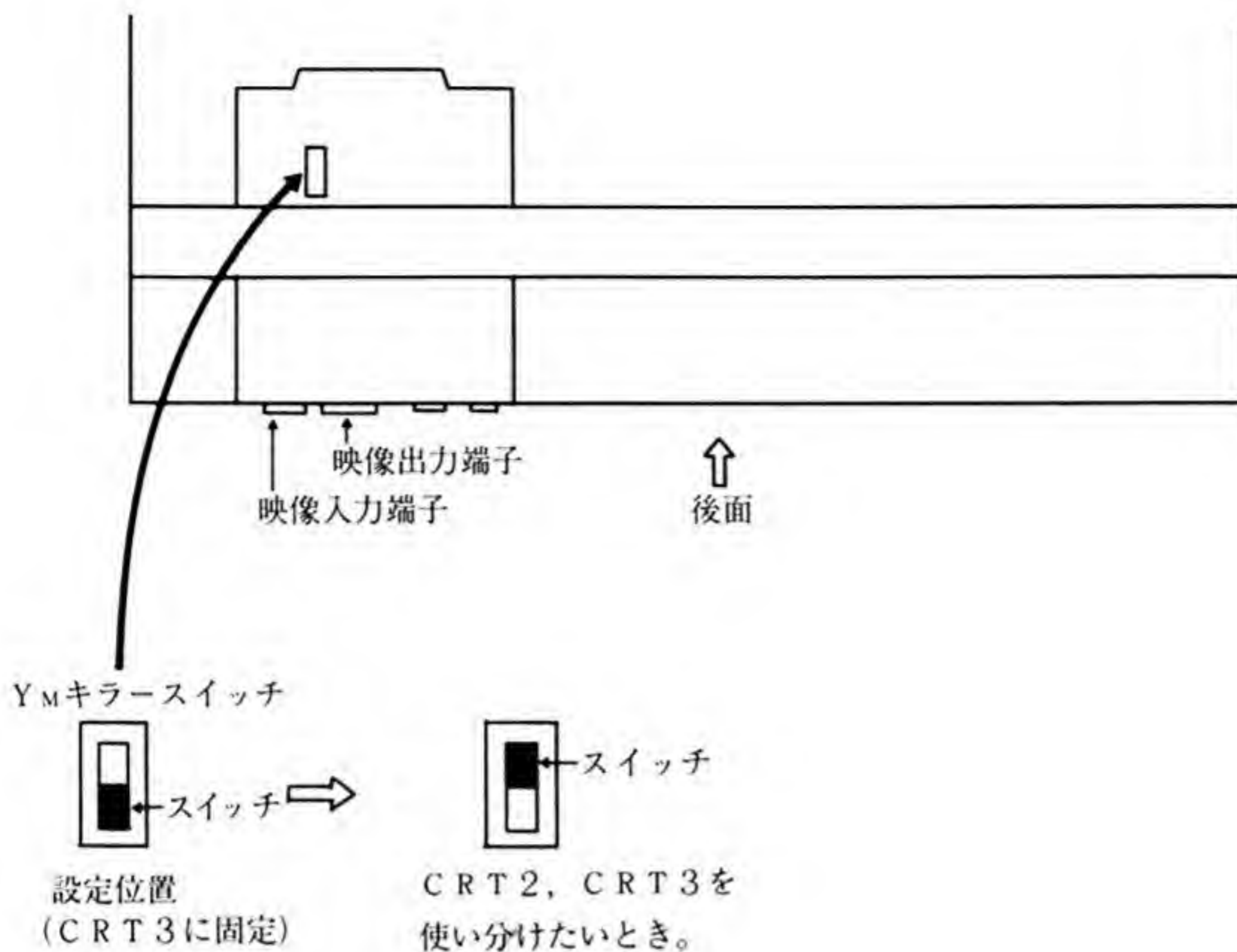
④映像入力端子にモニターの映像信号と異る、映像信号が入力されている場合スーパーインポーズモードにしますとモニター画面が乱れます。モニターのスーパーインポーズ画面を使用するときは映像入力端子にはモニターと同一の映像信号を入れるかまたは何も接続しないでください。

⑤本機の映像入力端子へ入力される映像が白黒であれば、スーパーインポーズ時またはコンピューターモード時（インターレースモード）のときのコンピュータ画像は色がつかなかったり、部分的に着色したりすることがあります。ただし映像が白黒であってもカラーバースト信号がついていればカラーになります。

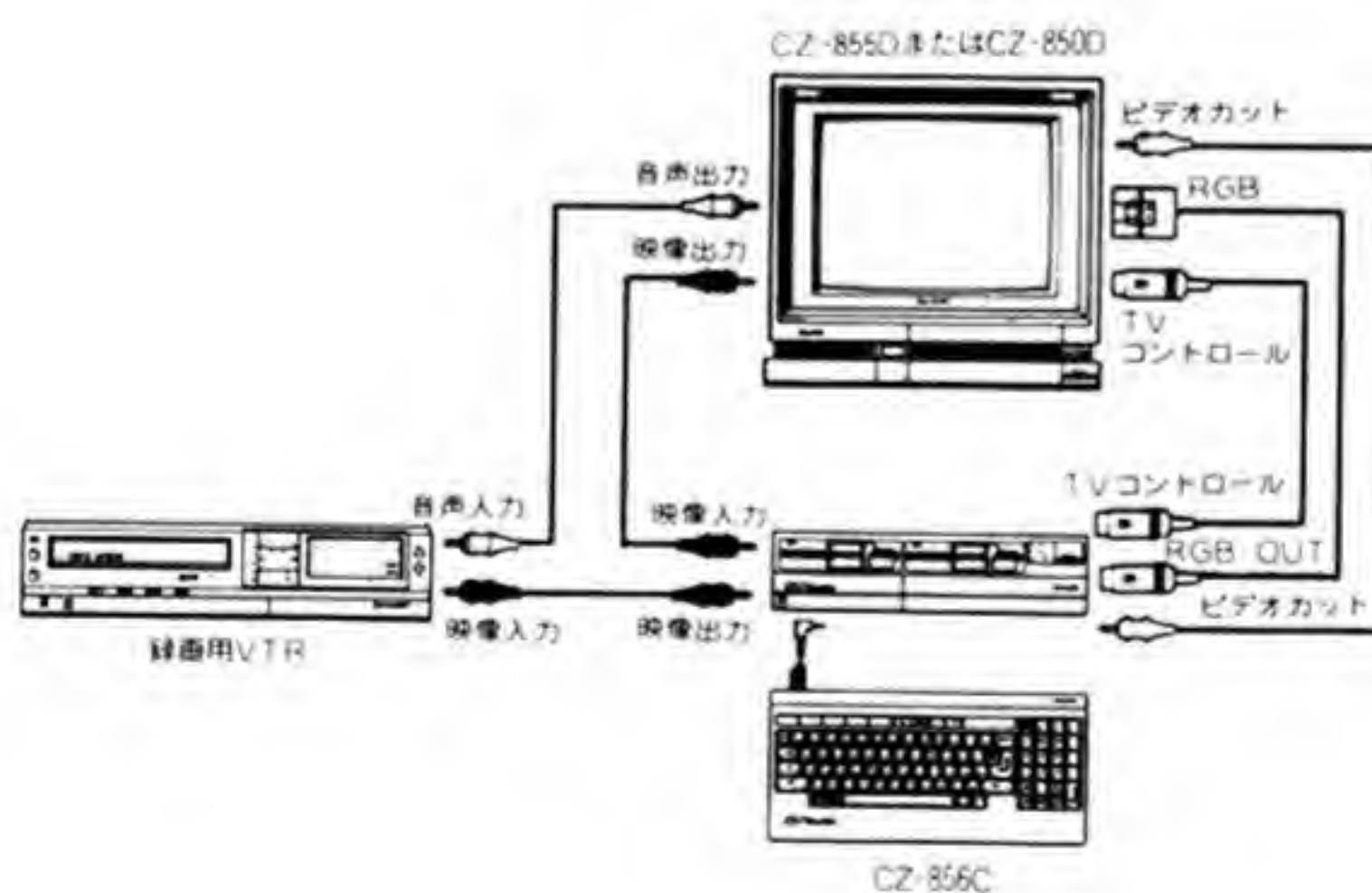
⑥本機に入力される映像信号の同期部分が劣化している場合、コンピュータ画像が乱れることがあります。たとえば、テレビのアンテナ入力信号が弱い場合のテレビ出力や幾度もダビングされたＶＴＲテープの再生信号等。

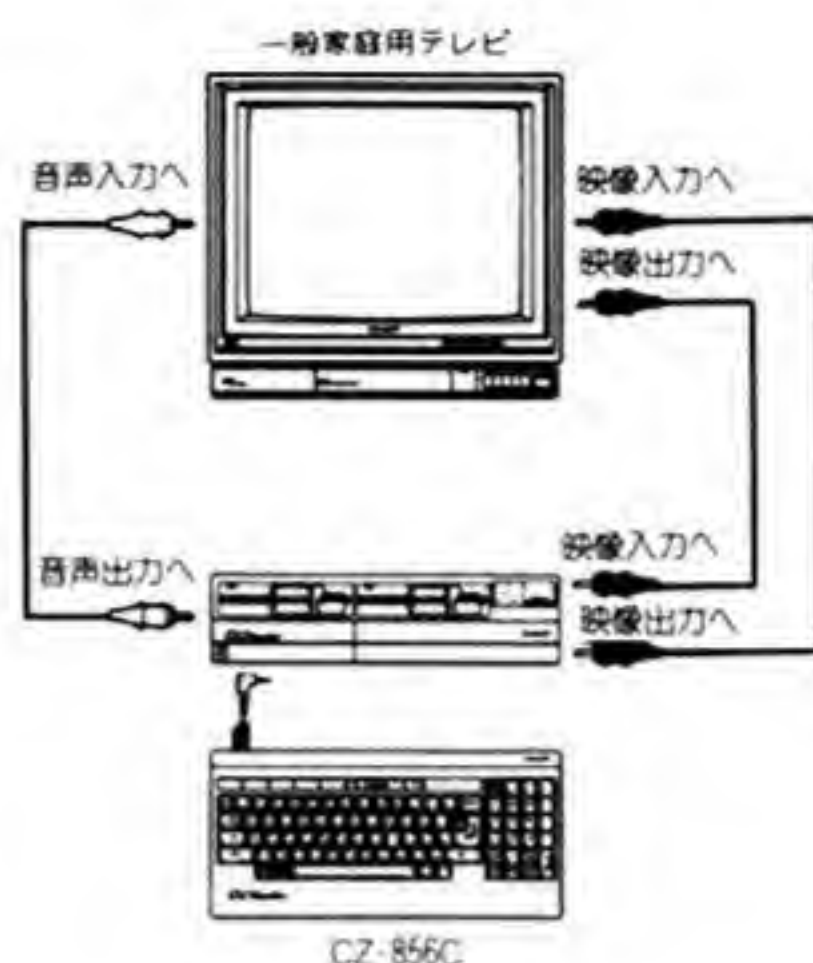
⑦次記のような接続では、画像が異常になるため使用しないでください。二重スーパーインポーズとなります。

(i)二重スーパーインポーズとなります。

(ii)専用ディスプレイテレビ（ＣＺ－８５５ＤまたはＣＺ－８５０Ｄ）をビデオモードにしたとき、同期流れがおこります。

⑧本機の映像出力端子より出力されるスーパーインポーズ画像はＢＡＳＩＣステートメントＣＲＴ３（テレビ放送とコンピュータ画面を同時に重ねて表示）を実行した状態になっています。プログラムにて映像出力をＣＲＴ２（テレビ放送のコントラストを下げて、コンピュータ画面を重ねて表示）状態とＣＲＴ３の状態とを使い分けたい場合は、上ぶたを開けて、Ｙｍキラースイッチをコンピュータ前面側へスライドさせてください。

参考

〈接続応用例①〉

• 専用ディスプレイテレビ（ＣＺ－８５５ＤまたはＣＺ８５０Ｄ）を使用して"コンピュータ画像""テレビとのスーパーインポーズ画像""テレビ画像"をＶＴＲに録画する場合。

〈接続応用例②〉

• 家庭用テレビ（ビデオ入出力端子付）に接続し、テレビ画像とコンピュータ画像を重ね合わせ、スーパーインポーズ画像を楽しむ場合。

〈ビデオマルチプロセッサ　CZ—8VP1との接続例〉

オプションのビデオマルチプロセッサ　CZ—8VP1は、A/Vスイッチャー、カラーコレクタ、ビデオエンハンサ、GENLOCK機能等を装備し、本機との組み合わせにより、コンピュータコントロールが可能な映像機器です。特長は次のとおりです。

①4入力3出力系と2入力1出力系のA／Vスイッチャーを独立に2基搭載しています。また、6系統の入力にはすべてブリッジ（共同接続）端子を設けました。

②オーディオ系は、すべてHi—Fiステレオ対応。また、左右独立のサウンドバランスつまみを装備しました。

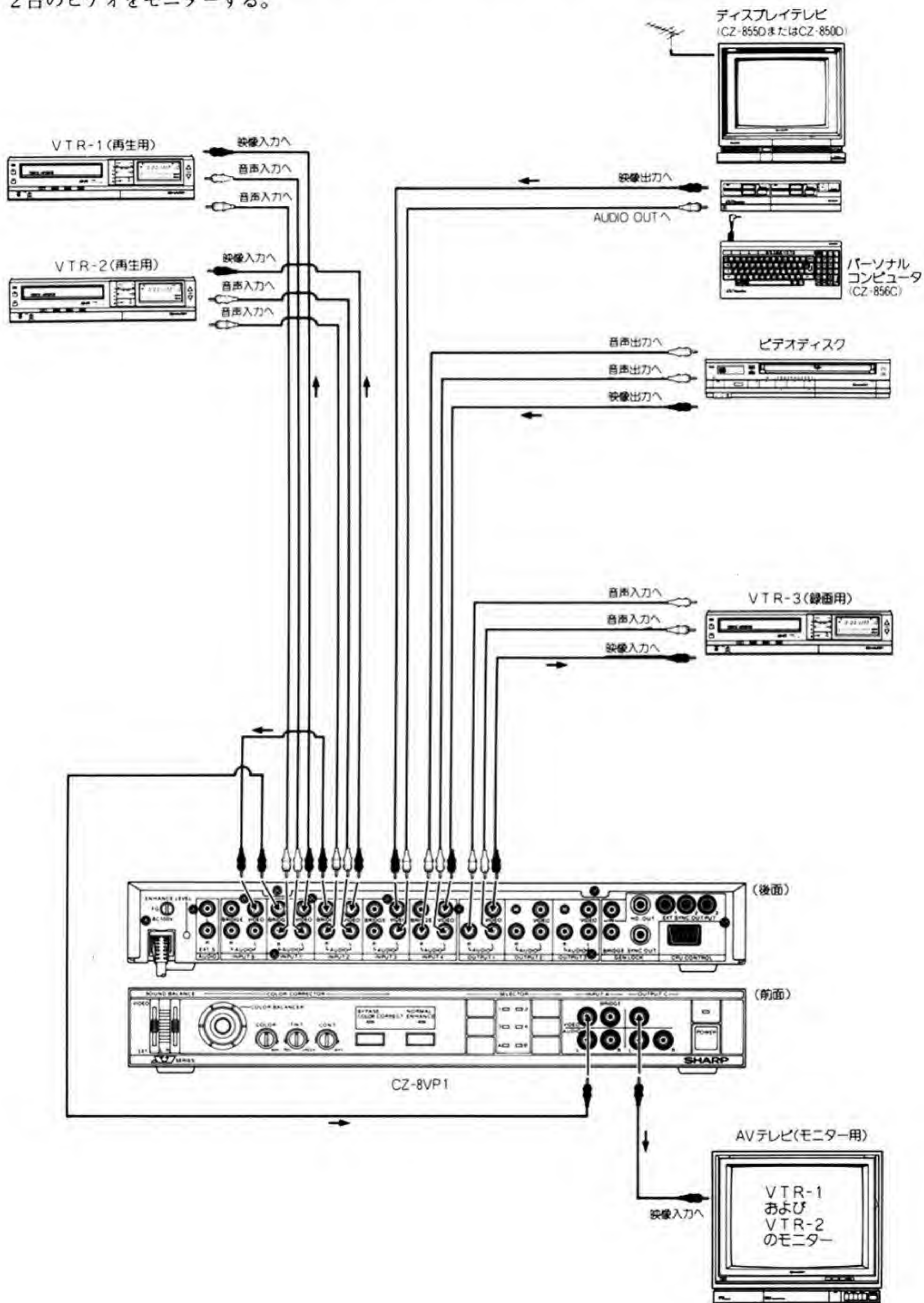
③映像のカラー信号を安定再生するカラーコレクター機能を装備しました。

④ディレーライン方式のビデオエンハンサ機能を装備しました。

⑤ABロール編集（複数のビデオを再生して編集）ができるゲンロック端子を装備しました。

⑥入力信号切換やカラーコレクタ、エンハンサのON／OFFがコンピュータでコントロールできます。

〈接続例〉

パソコン、ビデオ、ビデオディスク等、4つのソースを切り換えて録画し、またこれとは独立して2台のビデオをモニターする。

〈グラフィックツール・嬉楽画ターボ・ＣＺ―１１４ＳＦ〉

マウスを使って、コンピュータグラフィックスやタイトル文字の作成などを行うソフトウェアで、作成した画面をビデオ編集に利用することもできます。主な特長は次のとおりです。

①入力には、付属のマウスを使用します。

②メニューはわかりやすい絵文字（アイコン）表示。

③高解像度モード（タテ４００ライン）でも、解像度モード（タテ２００ライン）でも作画できます。ただし、作画をビデオ編集に利用する場合は、６４０×２００ドット／３２０×２００ドットのモードで作成してください。

④漢字を含む文字も簡単に表示できます。

⑤作画したデータをディスクに保存できます。

⑥秒単位に時間を指定して、作画を順次読み出して、ビデオ編集などに利用できます。

⑦作成した絵や文字をカラーインクジェットプリンタＩＯ－７００でコピーするためのユーティリティが付属しています。

付　　録

付録

A 1　テキスト画面(V-RAM)へのアクセス

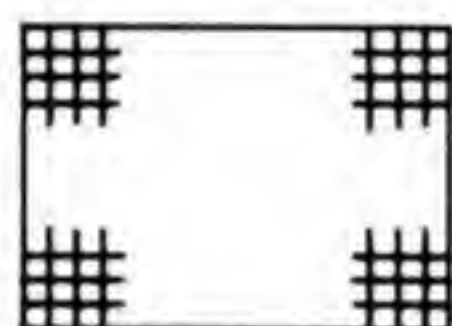
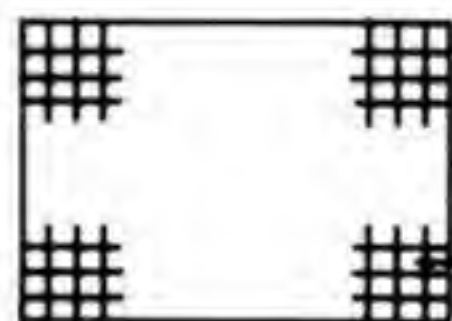
テキスト画面とその属性エリアは共にＩ／Ｏポートにあり、

　　テキスト画面が　＆Ｈ３０００～＆Ｈ３７ＦＦ（番地）
　　その属性が　　　＆Ｈ２０００～＆Ｈ２７ＦＦ（番地）
　　漢字属性が　　　＆Ｈ３８００～＆Ｈ３ＦＦＦ（番地）

の範囲で、それぞれ２ＫＢ（２０４８バイト）の容量をもっています。

テキスト画面とその属性エリアとはちょうど１対１に順序よく対応していて、次に示すステートメントや関数を使って１バイト単位でアクセスすることができます。

　　出力用：　ＰＯＫＥ＠、ＯＵＴ
　　入力用：　ＰＥＥＫ＠、ＩＮＰ

テキスト画面の属性１バイトのビット構成は下図のとおりです。

以下、下位ビットから順次説明していきます。

（１）　Ｂ、Ｒ、Ｇ………色の指定をする（⇨ＣＯＬＯＲ）

テキスト画面の文字の色はＢ（青）、Ｒ（赤）、Ｇ（緑）の３ビットにより次のように決定されます。

1　0　1 ………シアン（水色）

1　1　0 ………黄

1　1　1 ………白

（2）　CR………文字の反転モードの指定（⇨CREV）

テキスト画面の文字の反転の指定はビット3で行ないます。

ビット3が0のとき標準モード、1のとき反転モードとなります。

（3）　CF………文字の明滅モードの指定（⇨CFLASH）

テキスト画面の文字の明滅の指定は4ビットで行ないます。

ビット4が0のとき標準モード、1のとき明滅モードとなります。

（4）　CG………ROMCG／RAMCGの切り換えの指定（⇨CGEN）

キャラクタ・ゼネレータには、ROMCGとRAMCGの2種類があり、それぞれ256文字まで表示させることができます。

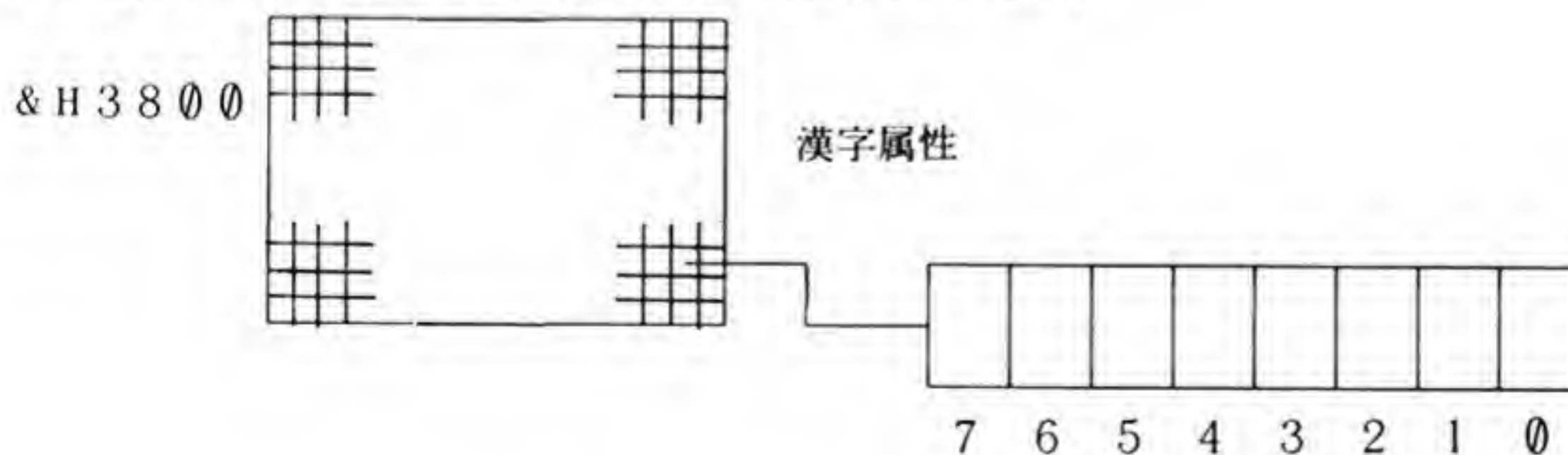
テキスト画面にROMCGで生成した文字を表示するか、RAMCGで生成した文字を表示するかの指定は、ビット5で行ないます。

ビット5が0のときROMCG、1のときRAMCGを指定します。

（5）　VS、HS…文字の拡大モードを設定します。（⇨CSIZE）

テキスト画面の文字のサイズの指定はビット6と7で行ないます。

ただし、ビット6（VS）に1を立てるときは、その横1行の属性ポートのビット6を立てなければなりません。

漢字属性1バイトのビット構成は次のようになっています。

以下、テキスト属性同様、下位ビットから順次説明していきます。

（1）　下位4ビット（ビット3，2，1，0）…漢字データ部の上位4ビットコードを示す。
　全角文字（漢字ＲＯＭ）が表示されているとき、漢字ＲＯＭアドレスの上位4ビットを示します。漢字ＲＯＭが選択されていない場合は無視されます。

（2）　ビット4………漢字ＲＯＭの第1水準か第2水準かの指定または、ＲＡＭＣＧのキャラクターモードか外字かの指定を行ないます。

（3）　ビット5…………アンダーラインの指定
　アンダーラインの指定は、ビット5で行ないます。
ここが0のときアンダーライン消法、1のときアンダーラインを表示します。

（4）　ビット6…………全角文字のサイド指定
　全角文字の場合ここがゼロならば文字の左側の部分、1ならば文字の右側部分であることを示しています。

（5）　ビット7…………ＲＯＭＣＧと漢字ＲＯＭの選択の指定

漢字属性のビット4とビット7、およびテキスト属性のビット4とビット7、およびテキスト属性のビット5（ＣＧ）の値によって、ＲＯＭＣＧ、ＲＡＭＣＧ、漢字ＲＯＭのうちどれかが選択されるかが、決まります。

テキスト属性 ビット5	漢字属性		
	ビット4	ビット7	
1	0	0	ＲＡＭＣＧ（キャラクタモード）
	0 1 1	1 0 1	ＲＡＭＣＧ（外字モード）
0	0 1	0	ＲＯＭＣＧ
	0	1	第1水準漢字ＲＯＭ
	1	1	第2水準漢字ＲＯＭ

A 2　組み込み関数以外の数学的関数

組み込み数値関数にない三角関数と双曲線関数およびそれらの逆関数を定義する公式を示します。使用する場合には各関数の定義域に注意する必要があります。

関数名	BASICの形式による公式	関数
secant	A (X) = 1／COS (X)	secx
consecant	B (X) = 1／SIN (X)	cosecx
cotangent	C (X) = 1／TAN (X)	cotanx
arcsine	D (X) = ATN (X／SQR (1－X^2))	$\sin^{-1}x$
arccosine	E (X) =-ATN (X／SQR (1－X^2)) ＋π／2	$\cos^{-1}x$
arcsecant	F (X) = ATN (SQR (X^2－1)) ＋ (SGN (X) －1) ＊π／2	$sen^{-1}x$
arccosecant	G (X) = ATN (1／SQR (X^2－1)) ＋ (SGN (X) －1) ＊π／2	$cosec^{-1}x$
arccotangent	H (X) =-ATN (X) ＋π／2	$cotan^{-1}x$
hyperbolic sine	I (X) = (EXP (X) －EXP (－X)) ／2	sinhx
hyperbolic cosine	J (X) = (EXP (X) ＋EXP (－X)) ／2	coshx
hyperbolic tangent	K (X) =-EXP (－X) ／ (EXP (X) ＋ EXP (－X)) ＊2＋1	tanhx
hyperbolic secant	L (X) =2／ (EXP (X) ＋EXP (－X))	sechx
hyperbolic cosecant	M (X) =2／ (EXP (X) －EXP (－X))	cosechx
hyperbolic cotangent	N (X) = EXP (－X) ／ (EXP (X) －EXP (－X)) ＊2＋1	cotanhx
arc-hyperbolic sine	O (X) = LOG (X＋SQR (X^2＋1))	$sinh^{-1}x$
arc-hyperbolic cosine	P (X) = LOG (X＋SQR (X^2－1))	$cosh^{-1}x$
arc-hyperbolic tangent	Q (X) = LOG ((1＋X) ／ (1－X)) ／2	$tanh^{-1}x$
arc-hyperbolic secant	R (X) = LOG ((SQR (1－X^2) ＋1) ／X)	$sech^{-1}x$
arc-hyperbolic cosecant	S (X) = LOG ((SGN (X) ＊SQR (X^2＋1) ＋1) ／X	$cosech^{-1}x$
arc-hyperbolic cotangent	T (X) = LOG ((X＋1) ／ (X－1)) ／2	$cotan^{-1}x$

以上の各関数を使うときは、「DEF　FN」ステートメントであらかじめプログラム中に定義しておくと便利です。これらの関数は、パラメータXの定義域に十分注意して使ってください。

```
(例)  100  DEF  FNA (X) =1／COS (X)
      ...  .......
      190  Y=...
      200  H=FNA (Y)
      ...  .......
```

A 3　数値精度の変換

数値は、整数の場合、－32768〜32767、実数の場合、約-1.7×10^{38}〜1.7×10^{38}で、単精度型のとき有効数字16けたの精度で表現されます。

数値は、変数の属性文字（%、!、#）、型変数換の関数（CINT、CSNG、CDBL）によって、その精度を変換することができます。

精度の変換は、次の規則に従って行なわれます。

(a) 数値が、別の精度の数値変数に代入されると、それは代入先の変数名の表わす精度で記憶されます。

(例)　A％＝3.14　………　A％は整数型変数なので、A％の値は3となります。

(b) 数値をそれより低い精度の変数に代入すると、数値は丸められて低い精度で記憶されます。

(例)　A！＝3.141592653 58＃　…　Aは単精度型変数なので、Aの値は3.1415927に丸められます。

(c) 数値をそれより高い精度の変数に代入すると、数値は高い精度の表示になりますが、より正確になることはありません。

(例)　A＃＝3.14　………　A＃は倍数精度型変数なので、A＃の値は3.139999999664724と表示されます。

(d) 式の値を求めるときは、最も精度の高いオペランドと同じ精度に変換されます。

(例)　A＃＝4＃／7　………　4＃が倍精度型数値なので、計算の結果は倍精度型となり、A＃の値は、.5714285714285714となります。

(e) 関数の値は、単精度8けたが保証されますが、引数に倍精度型の数値が含まれると倍精度16けたまで求めることができます。

(例)　A＝SQR（3）　………　Aの値は、1.7320508

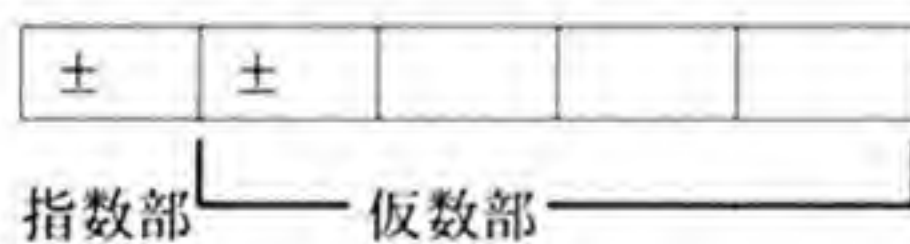
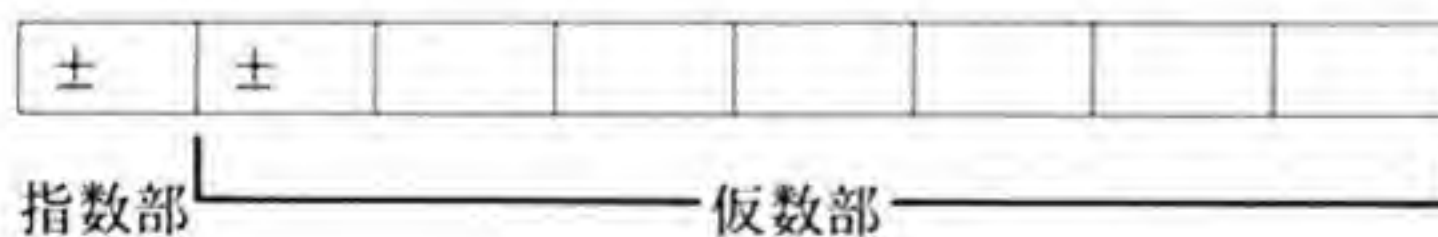
A＃＝SQR（3＃）　………　A＃の値は、1.7320508075688 77

(f) 数式の値の型変数を行なうには、CINT、CSNG、CDBLの各関数を使います。

CINTは値を整数型に、CSNGは単精度型に、CDBLは倍精度型にそれぞれ変換します。

(例)　A＝CINT（R／100）　………Rの値が314のとき、Aの値は3になります。

A 4　数値データの誤差

BASICで扱う単精度型および倍精度型の数値は、コンピュータ内部では2進浮動小数点形式で表現され、演算や比較もこの形式のまま行なわれます。

この内部表現上の制約のため、数値は必ずしも正確な値として記憶されるとは限らず、このために起こる、演算結果および関係式の比較における誤差、画面やライン・プリンタなどの出力装置に表示される値とのずれ、などに注意する必要があります。

たとえば、単精度型や倍精度型の数値を使った演算の結果が、数学では整数となる場合でも、必ずしも整数が得られるとは限りません。

数値データの内部表現の誤差が表示される値に及ぼす影響について、簡単な例をあげて説明してみましょう。

（例1）

```
10  X=0
20  FOR  I=0  TO  1500
30  PRINT  X;
40  X=X+.1
50  NEXT                          注）  .1=0.1
```

（例1）は、変数Xの初期値を0として0.1ずつ加えては、その値を表示するというプログラムです。

このプログラムを実行すると、はじめは、0、.1、.2 ………と増えていきますが、46.1を過ぎると、46.1、46.199999、46.299999、…というように誤差が現れます。

これは、内部表現の誤差がたまったために生ずる現象です。

この不合理を解消する方法として、次の3つが考えられます。

（1） 整数の1を加え、それを10で割る。

```
10  X=0
20  FOR  I=0  TO  1500
30  PRINT  X/10;
40  X=X+1
50  NEXT  I
```

（2） 10倍した値をCINTを使って整数型に変換してから10で割る。

```
10  X=0
20  FOR  I=0  TO  1500
30  PRINT  X;
40  X=X+.1
50  X=CINT(X*10)/10
60  NEXT  I
```

（3） STR$を使って文字型に変換してからVALを使って数値型に戻す。

```
10  X=0
20  FOR  I=0  TO  1500
30  PRINT  X;
40  X=X+.1
50  X=VAL(STR$(X))
60  NEXT  I
```

（例2）

```
10  FORI=0  TO  1  STEP  .1
20  PRINT  I;
30  NEXT  I
```

（例2）は、FOR、NEXTループにおいてループ変数の初期値を0、増分を0.1として、終了値1まで回すというプログラムです。

このプログラムを実行すると、0、0.1、0.2 ………、0.9となって、最後の1までループしません。

これも、内部表現の誤差によって生ずる現象です。

これを防ぐためには、次のように、ループ変数の増分を整数にします。

```
10  FOR  I=0  TO  10
```

```
２０　ＰＲＩＮＴ　Ｉ／１０;
３０　ＮＥＸＴ　Ｉ
```

（例３）

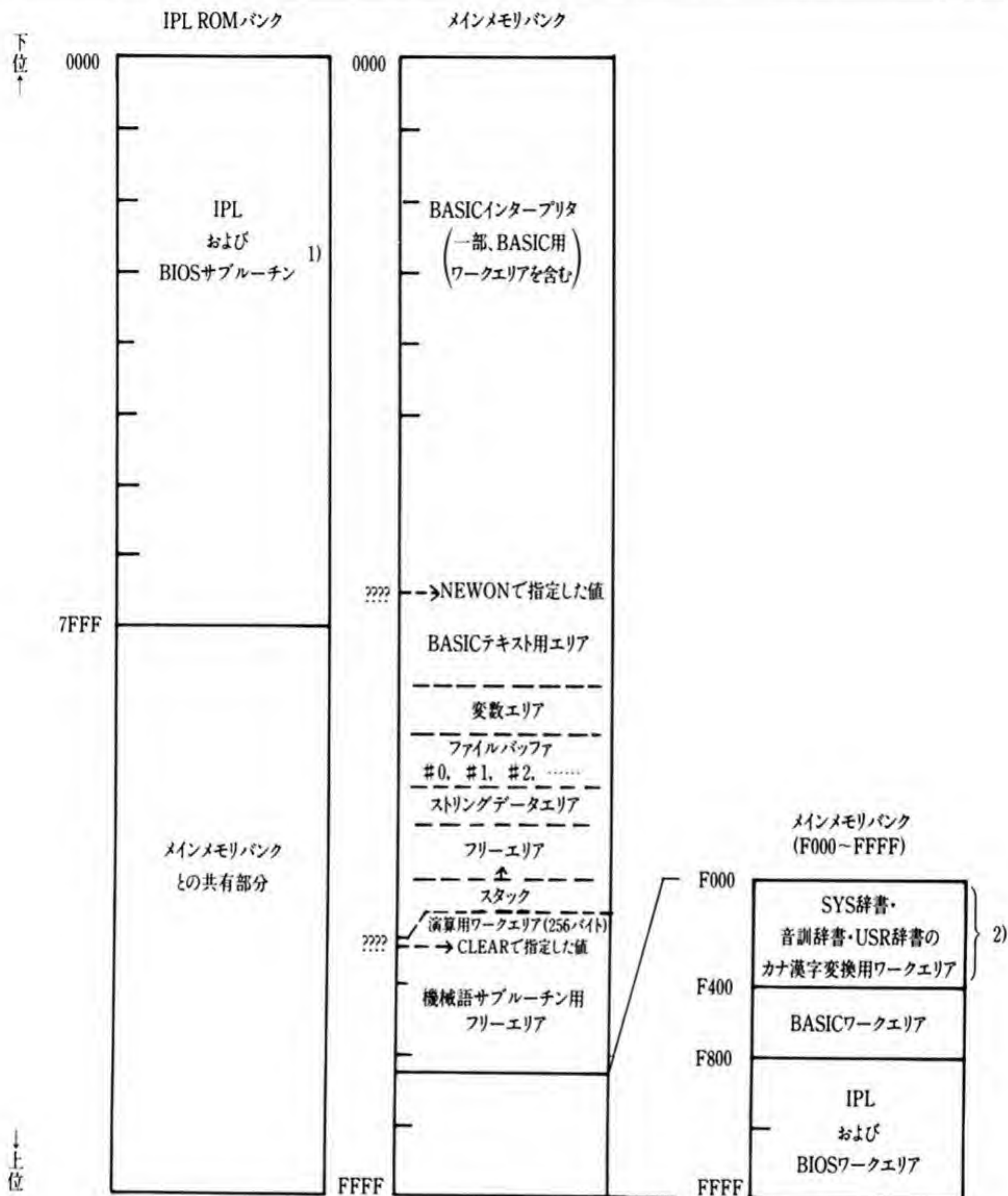
```
１０　Ｘ＃＝．１／３＊３
２０　Ｙ＃＝．１
３０　ＰＲＩＮＴＵＳＩＮＧ"＃＃＃．＃＃＃＃＃＃＃＃＃＃＃＃";Ｘ＃;Ｙ＃
４０　ＩＦ　Ｘ＃＝Ｙ＃　ＴＨＥＮ　ＰＲＩＮＴ"Ｘ＝Ｙ＃"
```

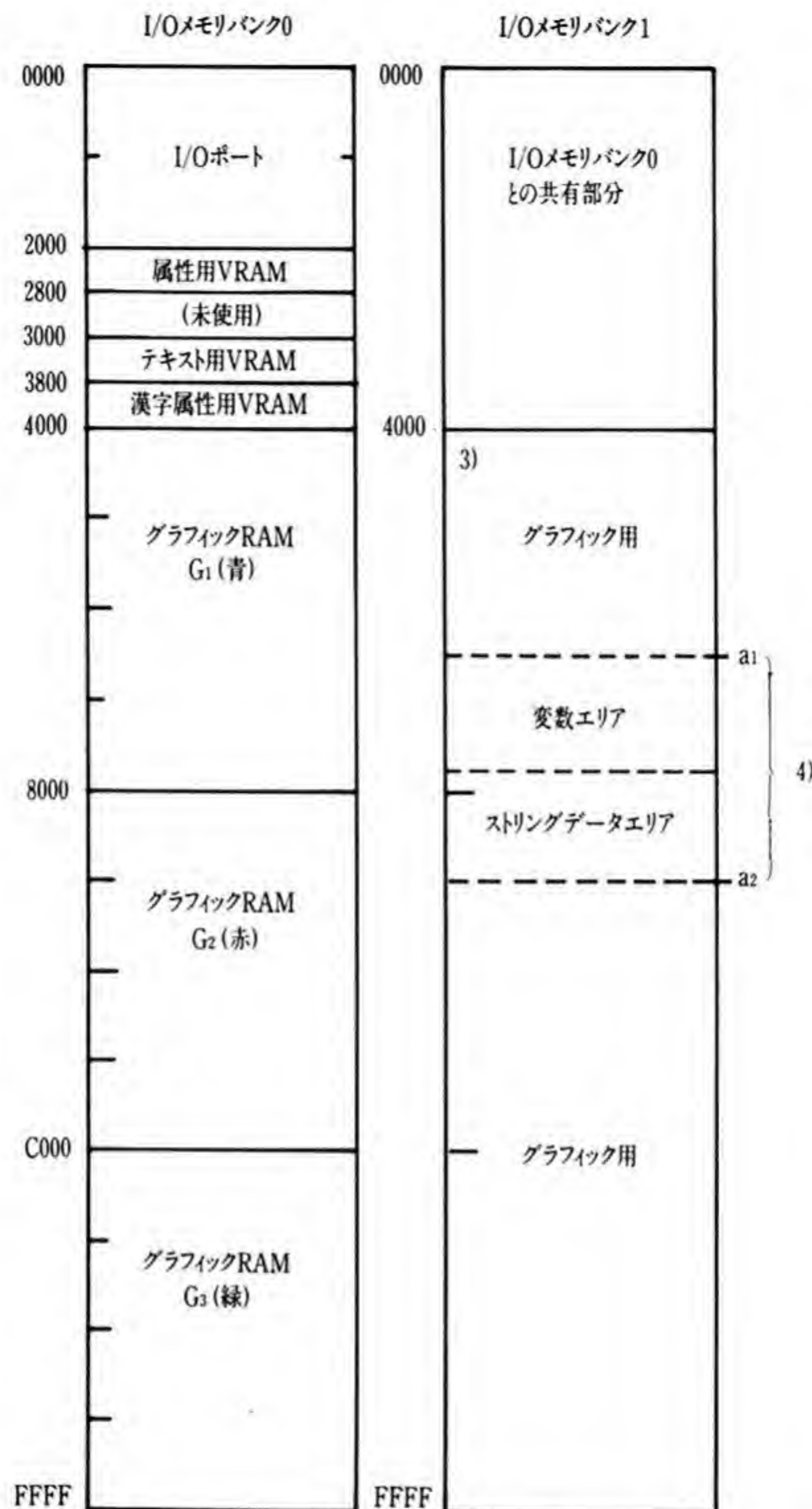
（例３）は、０．１を３で割って３倍した値をもつＸ＃と、０．１の値をもつＹ＃を比較して、等しければ「Ｘ＃＝Ｙ＃」と表示するプログラムです。

このプログラムを実行すると、「Ｘ＃＝Ｙ＃」と表示しません。これもまた、Ｘ＃とＹ＃との内部表現の誤差によって生じる現象です。

これを解消するためには、次のようにＩＦ文の論理式の部分を変えて、１０　の精度で比較するようにします。

```
１０　Ｘ＃＝．１／３＊３
２０　Ｙ＃＝．１
３０　ＰＲＩＮＴＵＳＩＮＧ"＃＃＃，＃＃＃＃＃＃＃＃＃＃＃＃";Ｘ＃;Ｙ＃
４０　ＩＦ　ＡＢＳ（Ｘ＃－Ｙ＃）＜．００００００００１　ＴＨＥＮ　ＰＲＩＮＴ
　　　"Ｘ＃＝Ｙ＃"
```

A 5 メモリマップ

1) ＢＩＯＳサブルーチンの機能
- キー入力
- 画面表示
- プリンタ出力
- カセット出力
- ディスク、ＨＤのリードライト
- グラフィックサブルーチン
- ＣＧの定義・読み込み
- 四則演算および関数
- その他

2) 辞書を使用しない場合は、
ＣＬＥＡＲ　＆ＨＦ４００を実行することにより、機械語サブルーチン用フリーエリアとして使用できます。

3)　ＯＰＴＩＯＮ　ＳＣＲＥＥＮ１および２の場合、ＶＤＩＭ変数エリアとして使用することができる。

ＯＰＴＩＯＮ　ＳＣＲＥＥＮ０が設定されている場合は、バンク０と同様グラフィック表示用エリアとして使用できる。

4)　ＶＤＩＭ変数エリアとして使用する場合、ＶＤＩＭ　ＣＬＥＡＲでその範囲を指定し、その外をグラフィック用として使用することができる。

a_1は変数エリアの先頭アドレス、a_2はエンドアドレス。ＶＤＩＭ　ＣＬＥＡＲを実行していなければ４０００～ＦＦＦＦが変数エリアとして使用できる。

A６　ユーザー定義文字の作りかた

英数字、カナ、セミグラフィック文字、特殊文字などの図形文字は、小さな点が集まって構成されています。１つの文字を構成する点は、半角文字で８×８または１６×８ドット、全角文字で１６×１６ドットあって、このドットパターンがＲＯＭＣＧ（Ｒead−Ｏnly Ｍemory Ｃharacter Ｇenerator）および漢字ＲＯＭに記憶されています。

これに対してＲＡＭＣＧ（Ｒandom Ａccess Ｍemory Ｃharacter Ｇenerator）があり、ユーザーが好きなドットパターンをデザインして、これに記憶させておくことができます。

ユーザーがデザインできる図形文字（以下、ユーザー定義文字といいます）には、そのサイズによって、

〔1〕　たて８×横８ドット
〔2〕　たて１６×横８ドット
〔3〕　たて１６×横１６ドット

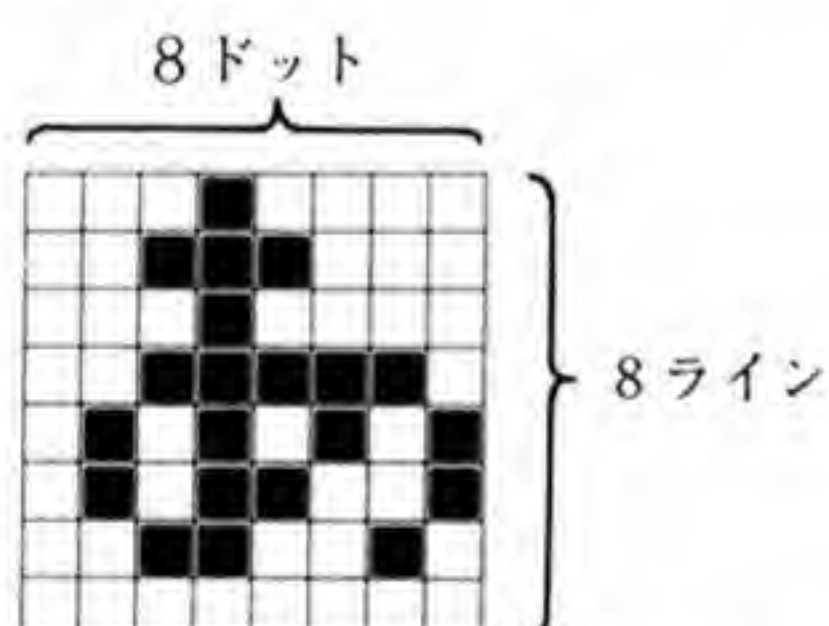
の３つのタイプがあります。

ここでは、その各々について作りかたと表示のしかたについて説明します。

〔1〕　たて８×横８ドットの場合

たて８×横８ドットのユーザー定義文字を定義するには、２４バイトの文字列が必要です。なぜならば、その文字列の表わすビットパターンが１個のユーザー定義文字を形成するからです。すなわち、１バイトのキャラクタコードのビットパターン８けたが、横８×たて１ドットパターンを表わし、８バイトの文字列でたて８ラインのドットパターンを構成しています。これが３原色の合成原理により、青、赤、緑の３画面分必要なので、結局、８バイト×３画面＝２４バイトの文字列が必要となります。

この２４バイトの文字列のうち、最初の８バイトが青、次の８バイトが赤、最後の８バイトが緑の部分のドットパターンを表わしています。

たとえば、ひらがなの「あ」という文字を青地にシアン色で作る場合についてみてみましょう。

■はドットがセットされていることを示す。

ユーザー定義文字は、８×８ドットパターンで表わされ、ひらがなの「あ」は上のようなパターンになります。これを２進数のパターンで表わし、さらに１６進数表現に変換すると、次のようになります。

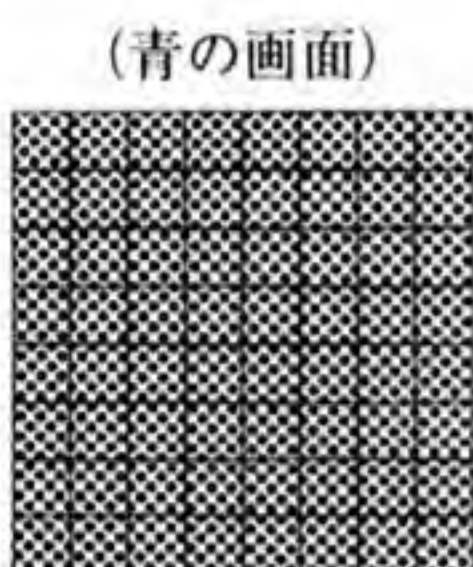
（２進数表現）		（１６進数表現）	
０００１００００	⇨	１０	……①
００１１１０００	⇨	３８	……②
０００１００００	⇨	１０	……③
００１１１１１０	⇨	３Ｅ	……④

0 1 0 1 0 1 0 1 ⇨ 5 5 ……⑤
0 1 0 1 1 0 0 1 ⇨ 5 9 ……⑥
0 0 1 1 0 0 1 0 ⇨ 3 2 ……⑦
0 0 0 0 0 0 0 0 ⇨ 0 0 ……⑧

したがって、このドットパターンは、

HEXCHR$("10 38 10 3E 55 59 32 00")
　　　　　① ② ③ ④ ⑤ ⑥ ⑦ ⑧

となります。

シアン色は青と緑を合成して得られ、青地にシアン色の「あ」という文字を作るためには、青の画面、赤の画面、緑の画面に対するパターンをそれぞれ次のように定義しなければなりません。

(青の画面)		(赤の画面)		(緑の画面)	
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 0 0 1 0 0 0 0	⇨1 0
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 0 1 1 1 0 0 0	⇨3 8
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 0 0 1 0 0 0 0	⇨1 0
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 0 1 1 1 1 1 0	⇨3 E
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 1 0 1 0 1 0 1	⇨5 5
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 1 0 1 1 0 0 1	⇨5 9
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 0 1 1 0 0 1 0	⇨3 2
1 1 1 1 1 1 1 1	⇨F F	0 0 0 0 0 0 0 0	⇨0 0	0 0 0 0 0 0 0 0	⇨0 0

したがって、青地にシアン色の「あ」を定義するのに必要な文字列は、

HEXCHR$ ("FFFFFFFFFFFFFFFF00000000000000001038103E55593200")

となります。

このサイズのパターンを定義できるコード（キャラクタコード）は0～255で、256個の文字を定義することができます。たとえばこれをコード32に定義するにはDEFCHR$ステートメントを使って、

DEFCHR$ (32) = HEXCHR$ ("FFFFFFFFFFFFFFFF00000000000000001038103E55593200") と書きます。

こうして定義した文字は、CGEN1とPRINT#0ステートメントを使って画面に表示することができます。

(例)
```
100 KMODE 0
110 CGEN1
120 A$ = "FFFFFFFFFFFFFFFF00000000000000001038103E55593200"
130 DEFCHR$ (32) =HEXCHR$ (A$)
140 PRINT#0, CHR$ (32)
150 CGEN0
```

青の画面、赤の画面、緑の画面の３枚とも同じデータの場合は、８バイトの文字列ですませることができます。

たとえば、黒地に白の「あ」を定義するには、

DEFCHR＄（32）＝HEXCHR＄（"1038103E555932001038103E555932001038103E55593200"）

としないで、

DEFCHR$（32）＝HEXCHR＄（"1038103E55593200"）

とするだけですみます。

〔2〕　たて１６×横８ドットのユーザー定義文字の定義には、８×８ドットの文字がたて２倍にのびたパターンなので、４８バイトの文字列が必要です。

１６×８ドットのユーザー定義文字は、１６×８ドットパターンで表わされます。ここにデザインしたパターンを８×８のときと同様に２進数のパターンで表わし、さらに１６進数表現に変換してコードの文字列にするのです。

このサイズのパターンを定義できるコードは＆Ｈ１００、＆Ｈ１０２、＆Ｈ１０４、………、＆Ｈ１ＦＥの１２８文字分です。たとえばあるパターンをコードの＆Ｈ１００に定義するには、

DEFCHR＄（＆H100）＝HEXCHR＄（"HHH………HHH"）
９６けた（４８バイト）

と書きます。

こうして定義した文字は、ＣＧＥＮとＰＲＩＮＴ＃０ステートメントを使って画面に表示することができます。

```
110　CGEN 2
120　A＄＝"HHH………HHH"　→４８バイトの任意の文字列
130　DEFCHR＄（＆H100）＝HEXCHR（A＄）
140　PRINT＃0，CHR＄（0）
```

150　CGEN 0

※　＆Ｈ１００のコードは　ＰＲＩＮＴ＃０，ＣＨＲ＄（０）
　　＆Ｈ１０２のコードは　ＰＲＩＮＴ＃０，ＣＨＲ＄（＆Ｈ２）
　　＆Ｈ１０４は　　　　　ＰＲＩＮＴ＃０，ＣＨＲ＄（＆Ｈ４）
　　………………………………

というように書いて表示します。

なお、青の画面、赤の画面、緑の画面の３枚とも同じデータの場合は、１６バイトの文字列ですませることができます。

〔3〕　たて１６×横１６ドットの場合

たて１６×横１６ドットのユーザー定義文字の定義には、９６バイトの文字列が必要です。

（青の画面）

1																	17
2																	18
3																	19
4																	20
5																	21
6																	22
7																	23
8																	24
9																	25
10																	26
11																	27
12																	28
13																	29
14																	30
15																	31
16																	32

１６×１６ドットのユーザー定義文字は、１６×１６のドットパターンで表わされます。ここにデザインしたパターンを、８×８のときと同様に２進数のパターンで表わし、さらに１６進数表現に変換してコードの文字列にします。

このサイズのパターンを定義できるコードは＆Ｊ７６２１～＆Ｊ７６６０の６４文字分です。たとえば、あるパターンをコードの＆Ｊ７６２１に定義するには、

DEFCHR＄（&J7621）＝HEXCHR＄（"HHH…HHH"）

１９２けた（９６バイト）

と書きます。

こうして定義した文字は、ＫＭＯＤＥ１を実行後、普通のＰＲＩＮＴ文を使って画面に表示することができます。

ＫＭＯＤＥ１

ＰＲＩＮＴ　ＣＨＲ＄（＆Ｊ７６２１）

なお、青の画面、赤の画面、緑の画面の３枚とも同じデータの場合は、３２バイトの文字列ですませることができます。

シャープ株式会社

本　　　　社	〒545	大阪市阿倍野区長池町22番22号 電話　06（621）1221（大代表）
電子機器事業本部	〒329-21	栃木県矢板市早川町174番地 電話　02874（3）1131（大代表）

お客様へ……お買いあげ年月日、お買いあげ店名を記入されますと、修理などの依頼のときに便利です。

お買いあげ年月日	年　　月　　日
お買いあげ店名	
	電話番号
もよりの お客様ご相談窓口	
	電話番号

TMAN-3614CEZZ

T0805-A　④